SINTRAN III VSX/500$ ND-570.146/CXA - HILDUR$

NORD SPOOLING SYSTEM/VALHALL III

OUTPUT FOR USER V S X

OF (SIN-J-LISTING:VSX)VSX-PART-2:LIST;1

GENERATED 16.25.33 DECEMBER 18, 1984

PRINTED 22.00.53 JANUARY 16, 1985

NORD SPOOLING SYSTEM/VALHALL III

```
    1  %FILE-NAME: Sintran III VSX
    2  %% :: Part Two (Superlisting)
$NONUMB


 SINTRAN III - VSX J  OBELIX *
*       ND 100/CX.1053.      *
*   PROD-DEV. OP-SYS & COMM. *

 --- SINTRAN III BATCH PROCESSOR ---

 USER AUX SINTRAN ENTERED AT 18.19.22      20 NOVEMBER    1984

MAXIMUM TIME IS     999 MINUTES
```

```
@CC MODE-FILE BUILD WITH SINGEN
@CC  **** N500.0000 SUPERLISTING VSX                L****

@CC
@CC  SYSTEM SERIAL NUMBER :
@CC
@CC  **** SIII-VSX VERSION J ****
@CC
@CC  ============================================================================

@DATCL
  18.19.24      20 NOVEMBER   1984
@CC
```

```
@CC

@CC  ------------------------------------------------------------

@CC  % (AUX)SEMA1 % (AUX)S3VS-1 % (AUX)BOUT-1 % (AUX)LIST-01        #

@CC  % 7ENDC<9POFS, IF VSX: 9EMRE<174000, MCCLC<114000              #

@CC  % 9EMRE<9EIOB<177000                                           #

@CC  ------------------------------------------------------------

@CC

@(AUX)SEMCHK

@CC %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

@CC  THIS LISTING CONTAINS THE CONFIGURATION-

@CC  DEPENDENT PART OF YOUR SINTRAN-III LISTING.

@CC %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


@CC ------------------------------------------------------------------------

@CLOSE 100
@OPEN (SYNTAX:EX-SCRATCH)SCRATCH:DATA,WX
FILE NUMBER IS 000100
@SET-PERM-OPEN 100B
@CC ------------------------------------------------------------------------


@CC  GET DATE INTO SINTRAN -------------------------

@MAC

- MAC
XXX=12000;YYY=400;)9TABL XXX YYY YYY
100/063000  LDA  (PAR
    MON 113
    JMP I (177773
)FILL
PAR,CLDAT
200/000051  CLDAT,0;0;0;0;0;0;0
)CLEAR
)9EXIT
@GO 100

0<170
)ZERO
)9ASSM (SIN)EX-SLIST
 **** 000000 DIAGNOSTICS ****

GENDA:000055
GENDA/000000  77MIN;77HOU;77DAY;77MON;77YEA;
202/000023  77MIN=^
203/000022  77HOU=^
```

```
204/000024   77DAY=^
205/000013   77MON=^
206/003700   77YEA=^
GENDA<GENDA+4
)PRINT
000055/000023
000022
000024
000013
003700
@
)9ASSM (,,LOLIST:SYMB
)LIST
)9EXIT
@CC ----------------------------------------------------


@CC DUMPING AND MIXING (AND ASSEMBLING) OF THE VARIABLE

@CC PARTS OF SINTRAN


@CONNECT-FILE (SIN)CX-BPUN3:SYMB,103,RX
@CONNECT-FILE (AUX)S3VS-1:SYMB,105,WA
@FMAC

- MACF -
IMAGE-FILE :
XXX=20000;YYY=400;ZZZ=10000;)9TABL XXX YYY ZZZ
1,0,103$

)9READ
)9READ
)9ASSM (AUX)LOLIST
 **** 000000 DIAGNOSTICS ****

GENDA/030060    77MIN;77HOU;77DAY;77MON;77YEA
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                                          %%
%%  SYSTEM CONFIGURATION IS DEFINED BELOW.  %%
%%                                          %%
%%  THE FOLLOWING FEATURES WILL ALWAYS BE   %%
%%  INCLUDED:                               %%
%%  -CARRIGE DELAY, RESTART                 %%
%%  -USER DEFINED ECHO AND BREAK            %%
%%  -ONE BATCH PROCESSOR                    %%
%%  -CHARACTER-CONVERTING ROUTINE LPPUT     %%
%%  -RT-PROGRAM-LOG, CPU-HISTOGRAM.         %%
%%  -2 INTERNAL DEVICES AND 5 SEMAPHORES    %%
%%                                          %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% START LIBRARY MARKS


175777/000000  8REST MAGTP 8ECBR 8LPPU TELIX CRDLY 8LOG 8HIST 8BCH1 IND01 IND02 8SM05 77XPO

175777/012725  8BDIS 8BD1 7B1U0
175777/000000  99SM1 8MT1 8BFD1
175777/000000  8M1U0 8F1U0
175777/000000  8TR5 8BP5 8TR6 8BP6
175777/000000  8CXHD          % COMMON HDLC LIB.-MARK
175777/000000  8C1HD 8C2HD
175777/000000  8PIOC 8PC01
175777/000000  8GPIO
175777/000000  8OCTO OCTOO
175777/000000  8MPM4
175777/000000  8QBPO          % TPS FOR X21
175777/000000  8C1X2
175777/000000  8SM01
175777/000000  8UDMA 8UD01 8F5UD
175777/000000  8DLP1 8COSP
175777/000000  SLP1
SLD1=00005
8RTN=0027
175777/000000  IND01 IND02
175777/000000  IBL01
175777/000000  8BCH1
175777/000000  CXCPU          % SINTRAN VSX
175777/000000  8BU0      % 1 DEVICE BUFFERS
175777/000000  8NSDB 8DB00 8DB01
175777/000000  8RFAC 8SG01    %  1 FILE-ACCESS SEGMENTS
175777/000000  8ESCX          % EXTENDED ESCAPE HANDLING
175777/000000  8IBRS          % EXTENDED ISIZE MON-CALL
175777/000000  8ADAD 8AD01    %  1 TERMINAL ACCESS DEVICES
175777/000000  8N500          % NORD-500
175777/000000  8PR01          %  1 N500 PROCESS DESC
175777/000000  8ACC           % CPU ACCOUNTING FOR RT PROGRAMS
175777/000000  8IOAC          % BLOCK I/O ACCOUNTING
175777/000000  8PACL          % PANEL CLOCK
175777/000000  8SIBA
CCNO=01                       % RT-COMMON SIZE
175777/000000  8BACS
175777/000000  8SWLG 8DILG 8MCLG 8SREE 8LAMU 8RSEG 8QBPO
8SGN=045                    % USER SEGMENTS
 9POFS 106000
```

```
175777/000000  LISTZ
)DEC
33CPU=500
33CPN=5000
)OCT
175777/000000  8EXAD
175777/000000  99ENS 8XMSG 8ECBR 8XON 8STRN 8MOLI
175777/000000  8ALME
NALME=5
LBSEG=1000
%% END LIBRARY MARKS


                        %% SYSTEM CONFIGURATION ACCORDING TO LIBRARY-MARKS:
)9ASSM (SIN)PRE-GEN-LIB
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)GEN-LIB,1                  %        UNSORTED LIBRARYMARKS SINTRAN-III VERSION J
                    %                                                          LOG UNIT NO
 "8N500 CXCPU;                % SINTRAN III VSX/500
 "8BDIS;              % BIG DISC DRIVER 37,75,288,30,60,90,150,3X75 MB
 "8BD1;               % CONTROLLER 1 BIG DISC 37,75,288,30,60,90,150,3X75 MB
 "7B1U0;              % UNIT 0, 75 MB, CONTROLLER 1
 "99SM1;              % STC MAG-TAPE DRIVER CONTROLLER 1                      526,560
 "8MT1;               % LINE DRIVER MAG-TAPE, CTRL 1
 "8M1U0;              % UNIT 0 MAG-TAPE, CONTROLLER 1                         40
 "8BFD1;              % BIG FLOPPY DISC DRIVER, CONTROLLER 1                  1145
 "8F1U0;              % UNIT 0 FLOPPY DISC, CONTROLLER 1 ,                    1000,1150
 "8TR5;               % DATA FIELD, TERMINAL 5                                44
 "8TR6;               % DATA FIELD, TERMINAL 6                                45
 "8BP5;               % BACKGROUND PROGRAM, TERMINAL 5                        44
 "8BP6;               % BACKGROUND PROGRAM, TERMINAL 6                        45
 "-LBSEG; LBSEG:001000     % = BACKGROUND SEGMENT SIZE (100000 = 128 K)
 "8BACS;              % BACKGROUND ALLOCATION SYSTEM
 "8SM01;              % SYNCHRONOUS MODEM 1                                   6
 "8XON;               % XON / XOFF, GIVES 8RON READER ON/READER OFF
 "8CXHD;              % GENERAL LIBRARY MARK HDLC DMA
 "8C1HD;              % GENERAL HDLC DMA DRIVER NO 1                          1360,1361
 "8C2HD;              % GENERAL HDLC DMA DRIVER NO 2                          1362,1363
 "8UDMA+8VICO;        % NEW UNIVERSAL DMA / VICOM DRIVER                      2100-2117
 "8F5UD;              % FAST  MON UDMA  FROM ND-500
 "8UD01;              % DATA FIELD NO 1  UDMA  (ND852)                        2100
 "8GPI0;              % GPIB INTERFACE 1
 "8OCTO;              % OCTOBUSS DRIVER
 "OCT00               % OCTOBUSS DATAFIELD FOR INTERFACE 1                   2400b - 2417b (slot no 20b - 37b)
 "8DLP1;              % LINE PRINTER 1                                        5,1167
 "SLP1;               % SPOOLING PROGRAM 1                                    1136
 "8COSP;              % COSMOS SPOOLING
"-SLD1; SLD1:000005        % = LOGICAL NUMBER SPOOLING INDEX 1
"-8RTN;  8RTN:000027       % = NUMBER OF RT DESCRIPTIONS, DEFAULT 24 (DEC)
"-8SGN;  8SGN:000045       % = NUMBER OF SEGMENT DESCRIPTIONS, DEFAULT 40 (DEC)
 "8SM05;              % 5 SEMAPHORES                                          300-304
 "IND01;              % INTERNAL CHARACTER DEVICE NO 1                        200
 "IND02;              % INTERNAL CHARACTER DEVICE NO 2                        201
 "IBL01;              % INTERNAL BLOCK DEVICE NO 1                            200,1210,1272
 "8BCH1;              % BATCH PROCESSOR NO 1                                  1236
 "8ALME.              % MON 61, FIXC500
"-NALME; NALME:000005      % = NO OF CONCURRENT ALLOCATED AREAS FOR FIXC500, DEFAULT 5
```

```
 "99ENS;          % MON 157/160 ENTSEG/FIXC
 "-CCNO; CCNO:000001      % = NUMBER OF K WORDS CORE COMMON (RT COMMON), DEFAULT 0
 "8N500;          % ND-500, GIVES 10 PROC. DESCR., FIXC500 AND INSTR/OUTST
 "8EXAD;          % EXTENDED ADDRESS MODE, MEMORY GREATER THAN 1 MB
 "CXCPU;          % CX CPU
 "8PRO1;          % 1 ND-500 PROCESS DESCRIPTION
 "8RFAC;          % REMOTE FILE ACCESS
 "8LAMU;          % LAMU
"8C1X2;          % UNIT 1 X-21 DRIVER                                        1374
 "8SIBA;          % 1 TO 3 SIBAS, MON SIBAS, 2 K RTCOMMON
 "8QBPO;          % TPS, TRANSACTION PROCESSING SYSTEM
 "8PIOC;          % PIOC
 "8PC01;          % UNIT 1 PIOC
 "8BUO;           % 1 K WORD DEVICE BUFFER
 "-9POFS; 9POFS:106000    % = START PAGING OFF AREA, DEFAULT 110000 (OCT)
"8ADAD;          % COMMON LIBRARY MARK FOR COSMOS
 "8AD01;          % TERMINAL ACCESS DEVICE NO 1, COSMOS
 "8PACL;          % AUTOMATIC UPDATE SINTRAN AND PANEL CLOCK ND-100
 "8SREE;          % SPECIAL MON REENT. SREEN
 "8STRN;          % MON 161/162, INSTR/OUTST
 "8RSEG;          % MON 53, READ SEGMENT TABLE ENTRY
 "8NSDB;          % MON 204/205 FOR NORD SYMBOLIC DEBUGGER
 "8DB01;          % 1 TERMINAL TO USE NORD SYMBOLIC DEBUGGER
 "8MOLI;          % MON LOGIN
 "8ESCX;          % MON 302/303 TURN ON/OFF DELAYED ESCAPE HANDLING
 "8IBRS;          % MON 313 NO OF CHAR IN INPUT BUFFER, CHAR UNTIL BREAK COND
 "8ACC;           % CPU ACCOUNTING FOR RT PROGRAMS
 "8IOAC;          % I/O ACCOUNTING FOR NUMBER OF DISC ACCESSES
 "8MPM4;          % MULTIPORT MEMORY 4
 "8LOG;           % RT-PROGRAM-LOG                                          1202,1352
 "8HIST;          % HISTOGRAM                                               1203
 "8REST;          % RESTART ROUTINE
 "8LPPU;          % CHARACTER CONVERTING ROUTINE
 "TELIX;          % TTY DRIVER TELIN/TELENT
 "CRDLY;          % CARRIAGE DELAY
"8ECBR;          % USER DEFINED ECHO AND BREAK TABLE 7
"-33CPU; 33CPU:000764     % = CPU TYPE
"-33CPN; 33CPN:011610     % = CPU NUMBER
 "
)LINE
 **** 000000 DIAGNOSTICS ****

1,0$

)9ASSM (SIN)POST-GEN-LIB
 **** 000000 DIAGNOSTICS ****


LOADR=200


)9RLPL
)9SCLC
"LISTZ
)9ASSM (SIN)MACROES-SIN1-GEN,105
 **** 000000 DIAGNOSTICS ****

"-LISTZ)9ASSM (SIN)MACROES-SIN1-GEN"
)9ASSM (SIN)SIN1-GEN
 **** 000000 DIAGNOSTICS ****
```

```
)9ASSM (SIN)MACROES-SIN2-GEN
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)SIN2-GEN
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)MACROES-SIN3-GEN
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)SIN3-GEN
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)MACROES-SIN4-GEN
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)SIN4-GEN
 **** 000000 DIAGNOSTICS ****

)9RLPL
)9SCLC
)9ASSM (SIN)PROCESS-IO
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)FENTER-FLEAVE
 **** 000000 DIAGNOSTICS ****

)9RCLC
)9SLPL
)9ASSM (SIN)CX-0B2
 **** 000000 DIAGNOSTICS ****

1,0$

175777/120574  MCSTA
)SYSDF
%% POSSIBLE FAULT (NOT FAULT) ERROR COMES NOW.
)9ASSM (SIN)FILELIST
% **** ERROR AT: 176000 **** POSSIBLE FAULT 110001

 **** 000001 DIAGNOSTICS ****

)SYSDF
)9ASSM (SIN)RTL-LIST
 **** 000000 DIAGNOSTICS ****


)9ASSM TERM,,(AUX)NNET-01-S:BPUN
SNETS←ENETS
)BPUN ZRO

)9ASSM TERM,,(AUX)BPUN-01-6:SYMB
NAMTA←MCSTA-1
)BPUN ZRO

)9ASSM TERM,,(AUX)BPUN-01-8:SYMB
172000←173777
)BPUN ZRO

)9ASSM 1,,NNETULI-01:SYMB
```

```
)ULIST
)9ASSM (SIN)CX-ULISTX
 **** 000000 DIAGNOSTICS ****

1,0,0$

9POFS<-1
)ZERO
)9EXIT
@CONNECT-FILE (AUX)S3VS-1:SYMB,105,WA
@CONT


"LISTZ
)9ASSM (SINTRAN)CX-OB32,105
 **** 000002 DIAGNOSTICS ****

"-LISTZ)9ASSM (SINTRAN)CX-OB32"
)9ASSM (SINTRAN)CX-PIT3-PIT0-OBJ
 **** 000000 DIAGNOSTICS ****

)9RLPL
)9SCLC
)9ASSM (SIN)MRES-SIN2
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)M1DRIVERS
 **** 000000 DIAGNOSTICS ****

)9ASSM (SIN)M2DRIVERS
 **** 000000 DIAGNOSTICS ****

)9RCLC
)9SLPL
)9ASSM (SIN)BFDIS-DRIVER
 **** 000000 DIAGNOSTICS ****

)9ASSM (SINTRAN)CX-PIT3-POF-OBJ
 **** 000000 DIAGNOSTICS ****

)9RLPL
)9SCLC
"8BDIS+8GDIS
)9ASSM (SIN)BIG-DI-DR
 **** 000000 DIAGNOSTICS ****

"8WDIS)9ASSM (SIN)WINCHESTER-8"
)9ASSM (SIN)STC-DRIVER
 **** 000000 DIAGNOSTICS ****

"77XPO; 77XPO=0
"
77EMR=*, *+77XPO/173401
%% POSSIBLE FAULT (NOT FAULT) ERROR COMES NOW.
9EMRE=*
% **** ERROR AT: 172727 **** POSSIBLE FAULT 172024

% **** ERROR AT: 172727 **** POSSIBLE FAULT 172023

XCORM=*
```

```
@
 **** 000002 DIAGNOSTICS ****


)SYSDF
)9ASSM (SIN)FILELIST,0
 **** 000000 DIAGNOSTICS ****

?
8REST 8ECBR 8LPPU TELIX CRDLY  8LOG
8HIST 8BCH1 IND01 IND02 8SM05  8BD1
7B1U0 99SM1  8MT1 8BFD1 8M1U0 8F1U0
 8TR5  8BP5  8TR6  8BP6 8CXHD 8C1HD
8C2HD 8PIOC 8PC01 8GPIO 80CTO OCT00
8MPM4 8QBPO 8C1X2 8SM01 8UDMA 8UD01
8F5UD 8DLP1 8COSP  SLP1 IND01 IND02
IBL01 8BCH1 CXCPU  8BU0 8NSDB 8DB00
8DB01 8RFAC 8SG01 8ESCX 8IBRS BADAD
BAD01 8N500 8PR01  8ACC 8IOAC 8PACL
8SIBA 8BACS 8SWLG 8DILG 8MCLG 8SREE
8LAMU 8RSEG 8QBPO LISTZ 8EXAD 99ENS
8XMSG 8ECBR  8XON 8STRN 8MOLI 8ALME
8ZBDI 99MGT 99CAC  8RON 99ENS 8ALME
 8RON 8EXAD 8STRN  8MT1 8CP51 BADAD
DT010 DT033 DT066 DT014 DT021 DT023
DT045 MTDI2 MTDI3 MTDI4 TAD01 BFI01
ND852 UBUSY  UFIN UBUSY OCT00 START
SRTER DILRT 9BPTM STSLI URT01 S41FP
PT3FU ELON3 ELOF3 EL03F DLOF3 EUSE3
DUSE3 T207R IBR3S T206R 3MLOG PT3MB
PT3MC HMAGT M8INC B4WIC M8IBC 3GSGN
3GRTD 3GRTN X2STA X21TO X21IN X2S01
X2E01 XSPT3 3BRPN 3DEBU RTDTT NAMFL
3SSME 3SSME  POOL IMGET IMPUT IMPPT
IMGET  POOL IMPUT 3SPLR PFIXC 3MAPS
3MSIB IOBUT 77CBU XLPT3 XSPT3 FPS41
SYSEV INTDF FPS41 BIGD2 BIGD3 BIGD4
WIGDI WIGD2 DRFIE DRFI2 GIGDI GIGD2
QDLAY QDLAY DRFIE DRFI2 BIGD2 BIGD3
BIGD4 GIGDI GIGD2 WIGDI WIGD2 HMAGT
TMAGT HMAGT TMAGT XCAHI PT3UD UASTR
UADDR UD11T APIOC APDRI PWRIT SI011
SI010 SI02I SI020 SI03I SI030 SI04I
SI040 SI05I SI050 SI06I SI060  3PIT


)9EXIT
@CONNECT FILE (AUX)S3VS-1:SYMB,105,WA
@CONT

"LISTZ
)9ASSM (SINT)CX-BUF-SIZE,105
 **** 000000 DIAGNOSTICS ****

"-LISTZ)9ASSM (SINT)CX-BUF-SIZE"8ZBDI+8ZGDI
)9ASSM (SIN)BIG-DI-DR
 **** 000000 DIAGNOSTICS ****

"8ZWDI)9ASSM (SIN)WINCHESTER-8"-8ZBDI -8ZGDI -8ZWDI)9ASSM (SIN)SWAP-DRIVERS"
)9RCLC
```

```
)9SLPL
)9ASSM (SIN)CX-OBJ-START
 **** 000000 DIAGNOSTICS ****


)SYSDF
)9ASSM (SIN)CX-ESLIST,0,0
 **** 000000 DIAGNOSTICS ****

%%%
%%% PATCHES IN CORE RES. (NOT THE SEGMENTS) COULD BE PLACED BELOW
%%%
*:033603
98END:063600
7ENDC:075367
)KILL 98END;98END=7ENDC+
*:033603

9SMRE<9EMRE-1
)9ASSM TERM,,(AUX)BPUN-01-A:BPUN
)BPUN ZRO

)9ASSM TERM,,(AUX)XSIULIST-01:SYMB
)ULIST

)9EXIT
@CONNECT-FILE (AUX)S3VS-1:SYMB,105,WA
@CONTINUE

"LISTZ
)9ASSM (SINTRAN)CX-PIT3-OBJ,105
 **** 000000 DIAGNOSTICS ****

"-LISTZ)9ASSM (SINTRAN)CX-PIT3-OBJ"
)9ASSM (SINTRAN)CX-PIT3-PIT0-OBJ
 **** 000000 DIAGNOSTICS ****

)9ASSM (SINTRAN)CX-PIT3-POF-OBJ
 **** 000000 DIAGNOSTICS ****


)9ASSM TERM,0,(AUX)LO1IST:SYMB
)LIST

)SYSDF
)9ASSM (SINTRAN)FILELIST,0,0
 **** 000000 DIAGNOSTICS ****


)SYSDF
)9ASSM (SINTRAN)CX-ESLIST,0,0
 **** 000000 DIAGNOSTICS ****


0<98END
)9ASSM TERM,,(AUX)BPUN-01-5:SYMB
)BPUN ZRO

BGSYS<CONIX
)9ASSM TERM,,(AUX)BPUN-01-7:SYMB
```

```
)BPUN ZRO

XS3=XSPT3; XD3=0; XN3=XEPT3-XSPT3
)9MOVE XS3 XD3 XN3
)9ASSM 1,0,(AUX)BPUN-01-PIT3:BPUN
XD3<XD3+XN3
)BPUN ZRO

XSPT3:114000      XEPT3:136672

999EN<99END
)9ASSM TERM,,(AUX)BPUN-01-B:BPUN
)BPUN ZRO

)9READ (AUX)NNET-01-S:BPUN
)9ASSM (AUX)NNETULI-01:SYMB
 **** 000000 DIAGNOSTICS ****

)9EXIT
@CONT


)9ASSM TERM,,(AUX)NNET-01-S:BPUN
SNETS<ENETS
)BPUN ZRO

)9READ (AUX)BPUN-01-B:SYMB
88XD=64000
88XS=172000
88XN=1777
)9MOVE 88XS 88XD 88XN
)9EXIT
@CONTINUE


)9ASSM TERM,,(AUX)BPUN-01-B:SYMB
88XD<88XD 88XN
)BPUN ZRO
```

```
%
% ============================================================
%  LABELS DEFINING SINTRAN III ADDRESS LIMITS.
%  NOTE THAT SOME PAGES OUTSIDE LOWER 128 KBYTE AREA MAY BE USED
%  BY SINTRAN FOR COREMAP TABLE, DMA BUFFERS ETC.

00000:000000    GNSTA:034000                % CONF. INDEPENDANT PART IN RESIDENT
GNSTA:034000    7ENDC:075367                % CONF. DEPENDANT PART IN RESIDENT
110000:110000   MCCLC:110234                % SINTRAN FILESYSTEM PART
CSSL0:002034    99END:175676                % SINTRAN STARTPROGRAM
9POFS:106000    9SMRE:106000                % I/O BUFFER AREA
BPOOL:030330    EIOBU:030356                % I/O BUFFER DESCRIPTOR
EIOBU:030356    9ESWP:031737                % SWAP-DRIVER
9SMRE:106000    9EMRE:172727                % "PAGING-OFF" CODE AREA
9EMRE:172727    9EIOB:173417                % I/O BUFFER AREA
77EMR:172727    77XPO:000000                % START AND LENGHT OF USER AREA IN POF
XSPT3:114000    XEPT3:136672                % PIT3 SEGMENT

% MCCLC: END OF CONFIGURATION DEPENDENT PART OF FILESYSTEM.
%        MUST NOT EXCEED 114000.

% ============================================================
)9EXIT
@CONT

0<-1
)ZERO
0/000000  MTP;NTRY1;LSTBI
ZBDIS;ZGDIS;ZWDIS;ZDRUM;ZDISC
%
% ONE POSSIBLE FAULT COMES NOW (NOT FAULT1)
%
)SYSDF
)9ASSM (SIN)MACM-SYMB:SYMB
% **** ERROR AT: 000010 **** POSSIBLE FAULT 034663

% **** ERROR AT: 000010 **** POSSIBLE FAULT 034664

% **** ERROR AT: 000010 **** POSSIBLE FAULT 057575

 **** 000003 DIAGNOSTICS ****

)9ASSM (SIN)MACM-ULIST:SYMB
 **** 000000 DIAGNOSTICS ****

)9READ (SIN)MACM-1718:BPUN
"99DR+55DR -ZDRUMMSTYP=0"99CDK+55CDK -ZDISCMSTYP=2"3B1U0+6B1U0 -ZGDISMSTYP=3"4B1U0+7B1U0+2E1U0 -ZBDIS
MSTYP=4
"HE1U0+2B1U0+3D1U0+1E1U0 -ZBDISMSTYP=5"3C1U0+6C1U0+9C1U0 -ZBDISMSTYP=6"2D1U0 -ZBDISMSTYP=7"2W1U0+1W1U0 -ZWDISMSTYP=
)9ASSM 1,0,(AUX)MACM-01-1718:BPUN
MTP-300+(STBI
)BPUN NTRY1
%============================================================

)9EXIT
@FMAC

 MACR
IMAGE FILE :
)9READ (SIN)CX BPUEE:BPUN
```

```
)9ASSM (SIN)CX-XESIULIST,0,0
 **** 000000 DIAGNOSTICS ****

)SYSDF
)9ASSM (AUX)LO1IST,,
 **** 000000 DIAGNOSTICS ****

)9EXIT
@CONT


)9ASSM 1,0,(AUX)BPUN-01-E:BPUN
XD=76000
XN=13777
XS=110000
)9MOVE XS XD XN
XD<XD+XN
)BPUN ZRO
?


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
)9EXIT
@QED

QED 4.3

*R (AUX)LO1IST
41091 WORDS READ
*L/RELOA/
RELOA=171626
*D1,.-1
*M TO(0)
*W (AUX)LIST-01
16758 WORDS WRITTEN
*F


@CC TEST SIZE => MODE=0

@(AUX)J STAT-TEST
MODE: 0


9EIOB IS: 173417
9EIOB WITH RT-DESCRIPTIONS INCLUDED: 175162
@FMAC

  MACF
IMAGE FILE :
17577//000000  98END;BGSYS;CONTX
0<-1
)ZERO
)9READ (AUX)BPUN-01-5:SYMB
)9READ (AUX)BPUN-01-7:SYMB
)9ASSM (SINTRAN)CX-ULIST4
 **** 000000 DIAGNOSTICS ****

)SYSDF
)9ASSM (AUX)LO1IST
```

```
 **** 000000 DIAGNOSTICS ****

?
CXCPU 3SINV GNFLA 5MAU1 5MAU1 5MAU1
5MAU1 JETTY JETTY WFRQI WFRQI AIRDW
  ACM CAMAC    GL  IOXN ASSIG TRACB
WFRQI   DIW  DOLW   US5   US6   US7
JETMO JETM2 POEAR P1EAR JETM3 JETM4
JETM5 JETM6 T8INB MSYSU SVER3 ESCRE
RUBRE REMOT LOCAL STACO STOCO  LREM
CMSTA REPAS CLSTA LOCAL LEAV3 BUFIN


)9EXIT
@CONT

)9ASSM TERM,0,(AUX)BPUN-01-5:SYMB
0<98END
)BPUN ZRO
)9ASSM TERM,0,(AUX)BPUN-01-7:SYMB
BGSYS<CONTX
)BPUN ZRO
)9ASSM TERM,0,0
)CLEAR
0<-1
)ZERO
)9READ (AUX)BPUN-01-6:SYMB
)9READ (SINTRAN)CX-BPUN2:SYMB
)9ASSM (SINTRAN)CX-ULIST2:SYMB
 **** 000000 DIAGNOSTICS ****

)9ASSM (SINTRAN)9FILELIST
 **** 000000 DIAGNOSTICS ****

)SYSDF
)9ASSM (AUX)L01IST:SYMB
 **** 000000 DIAGNOSTICS ****

?
3SINV SGFTA GNFLA 5MAU1 5MAU1 5MAU1
5MAU1 NAMFL 3SSME 3SSME  POOL IMGET
IMPUT IMPPT IMGET  POOL IMPUT

)9ASSM TERM,0,(SINTRAN)BPUN2-01:BPUN
110000<FSCLD
)BPUN ZRO

110000:110000      FSCLD:173021

)CLEAR
0<-1
)ZERO
)9READ (SINTRAN)CX-BPUN2B:SYMB
)9ASSM (SINTRAN)CX-ULIST2B:SYMB
 **** 000000 DIAGNOSTICS ****

)9ASSM (SINTRAN)9FILELIST:SYMB
 **** 000000 DIAGNOSTICS ****

)SYSDF
```

```
)9ASSM (AUX)L01IST:SYMB
 **** 000000 DIAGNOSTICS ****

?
VLSTF

)9ASSM TERM,0,(SINTRAN)BPUN2B-01:BPUN
FSSTA<SSCLD
)BPUN ZRO

FSSTA:140000      SSCLD:170447


)CLEAR
0<-1
)ZERO
)9READ (SINTRAN)CX-BPUN2C:SYMB
)9ASSM (SINTRAN)CX-ULIST2C:SYMB
 **** 000000 DIAGNOSTICS ****

)9ASSM (SINTRAN)9FILELIST:SYMB
 **** 000000 DIAGNOSTICS ****

)SYSDF
)9ASSM (AUX)L01IST:SYMB
 **** 000000 DIAGNOSTICS ****

?


)9ASSM TERM,0,(SINTRAN)BPUN2C-01:BPUN
SPSTA<SPCLD
)BPUN ZRO

SPSTA:110000      SPCLD:137634

)CLEAR
0<-1
)ZERO
)9READ (SINTRAN)CX-BPUN1:SYMB
)9READ
)9ASSM (SINTRAN)9RTLLIST
 **** 000000 DIAGNOSTICS ****

)9ASSM (SINTRAN)CX-ULIST1:SYMB
 **** 000000 DIAGNOSTICS ****

)SYSDF
)9ASSM (AUX)L01IST:SYMB
 **** 000000 DIAGNOSTICS ****

?
RWRT4 RWRT5 RWRT6 RWRT7 RWRT8 SCOM1
RCOM1 SCOM2 RCOM2 SCOM3 RCOM3 SCOM4
SCOM5 SCOM6 SCOM7 SCOM8 SCOM9 RCOM4
RCOM5 RCOM6 RCOM7 RCOM8 RCOM9 FDRT1
FDRT2 FIXRT SPRT2 SPRT3 SPRT4 SPRT5
SPRT6 SPRT7 SPRT8 SPRT9 SPR10 SPR11
SPR12 SPR13 SPR14 SPR15 WRT10 WRT12
WRT14 WRT15 WRT16 WRT17 WRT18 WRT19
```

```
WRT21 WRT22 WRT23 WRT24 WRT25 WRT26
WRT27 WRT28 WRT29 WRT30 WRT31 WRT32
WRT33 WRT34 WRT35 WRT36 WRT37 WRT38
WRT39 WRT40 BAK02 BAK03 BAK04 BAK07
BAK08 BAK09 BAK10 BAK11 BAK12 BAK13
BAK14 BAK15 BAK16 BAK17 BAK18 BAK19
BAK20 BAK21 BAK22 BAK23 BAK24 BAK25
BAK26 BAK27 BAK28 BAK29 BAK30 BAK31
BAK32 BAK33 BAK34 BAK35 BAK36 BAK37
BAK38 BAK39 BAK40 BAK41 BAK42 BAK43
BAK44 BAK45 BAK46 BAK47 BAK48 BAK49
BAK50 BAK51 BAK52 BAK53 BAK54 BAK55
BAK56 BAK57 BAK58 BAK59 BAK60 BAK61
BAK62 BAK63 BAK64 BAK65 BAK66 BAK67
BAK68 BAK69 BAK70 BAK71 BAK72 BAK73
BAK74 BAK75 BAK76 BAK77 BAK78 BAK79
BAK80 BAK81 BAK82 BAK83 BAK84 BAK85
BAK86 BAK87 BAK88 BAK89 BAK90 BAK91
BAK92 BAK93 BAK94 BAK95 BAK96 BAK97
BAK98 BAK99 BK100 BK101 BK102 BK103
BK104 BK105 BK106 BK107 BK108 BK109
BK110 BK111 BK112 BK113 BK114 BK115
BK116 BK117 BK118 BK119 BK120 BK121
BK122 BK123 BK124 BK125 BK126 BK127
BK128 BCH02 BCH03 BCH04 BCH05 BCH06
BCH07 BCH08 BCH09 BCH10 BC011 BC021
BC031 BC041 BC051 BC061 BC071 BC101
BC111 BC121 BC131 BC141 BC151 BC161
BC171 BC201 BC012 BC022 BC032 BC042
BC052 BC062 BC072 BC102 BC112 BC122
BC132 BC142 BC152 BC162 BC172 BC202
BC013 BC023 BC033 BC043 BC053 BC063
BC073 BC103 BC113 BC123 BC133 BC143
BC153 BC163 BC173 BC203 BC014 BC024
BC034 BC044 BC054 BC064 BC074 BC104
BC114 BC124 BC134 BC144 BC154 BC164
BC174 BC204 BC015 BC025 BC035 BC045
BC055 BC065 BC075 BC105 BC115 BC125
BC135 BC145 BC155 BC165 BC175 BC205
BC016 BC026 BC036 BC046 BC056 BC066
BC076 BC106 BC116 BC126 BC136 BC146
BC156 BC166 BC176 BC206 BC017 BC027
BC037 BC047 BC057 BC067 BC077 BC107
BC117 BC127 BC137 BC147 BC157 BC167
BC177 BC207 BC018 BC028 BC038 BC048
BC058 BC068 BC078 BC108 BC118 BC128
BC138 BC148 BC158 BC168 BC178 BC208
BC019 BC029 BC039 BC049 BC059 BC069
BC079 BC109 BC119 BC129 BC139 BC149
BC159 BC169 BC179 BC209 RU011 RU021
RU031 RU041 RU051 RU061 RU071 RU101
RU111 RU121 RU131 RU141 RU151 RU161
RU171 RU201 RU012 RU022 RU032 RU042
RU052 RU062 RU072 RU102 RU112 RU122
RU132 RU142 RU152 RU162 RU172 RU202
RU013 RU023 RU033 RU043 RU053 RU063
RU073 RU103 RU113 RU123 RU133 RU143
RU153 RU163 RU173 RU203 RU014 RU024
RU034 RU044 RU054 RU064 RU074 RU104
RU114 RU124 RU134 RU144 RU154 RU164
```

RU174 RU204 RU015 RU025 RU035 RU045
RU055 RU065 RU075 RU105 RU115 RU125
RU135 RU145 RU155 RU165 RU175 RU205
RU016 RU026 RU036 RU046 RU056 RU066
RU076 RU106 RU116 RU126 RU136 RU146
RU156 RU166 RU176 RU206 RU017 RU027
RU037 RU047 RU057 RU067 RU077 RU107
RU117 RU127 RU137 RU147 RU157 RU167
RU177 RU207 RU018 RU028 RU038 RU048
RU058 RU068 RU078 RU108 RU118 RU128
RU138 RU148 RU158 RU168 RU178 RU208
RU019 RU029 RU039 RU049 RU059 RU069
RU079 RU109 RU119 RU129 RU139 RU149
RU159 RU169 RU179 RU209 CR011 CR021
CR031 CR041 CR051 CR061 CR071 CR101
CR111 CR121 CR131 CR141 CR151 CR161
CR171 CR201 CR012 CR022 CR032 CR042
CR052 CR062 CR072 CR102 CR112 CR122
CR132 CR142 CR152 CR162 CR172 CR202
CR013 CR023 CR033 CR043 CR053 CR063
CR073 CR103 CR113 CR123 CR133 CR143
CR153 CR163 CR173 CR203 CR014 CR024
CR034 CR044 CR054 CR064 CR074 CR104
CR114 CR124 CR134 CR144 CR154 CR164
CR174 CR204 CR015 CR025 CR035 CR045
CR055 CR065 CR075 CR105 CR115 CR125
CR135 CR145 CR155 CR165 CR175 CR205
CR016 CR026 CR036 CR046 CR056 CR066
CR076 CR106 CR116 CR126 CR136 CR146
CR156 CR166 CR176 CR206 CR017 CR027
CR037 CR047 CR057 CR067 CR077 CR107
CR117 CR127 CR137 CR147 CR157 CR167
CR177 CR207 CR018 CR028 CR038 CR048
CR058 CR068 CR078 CR106 CR118 CR128
CR138 CR148 CR158 CR168 CR178 CR208
CR019 CR029 CR039 CR049 CR059 CR069
CR079 CR109 CR119 CR129 CR139 CR149
CR159 CR169 CR179 CR209 CW011 CW021
CW031 CW041 CW051 CW061 CW071 CW101
CW111 CW121 CW131 CW141 CW151 CW161
CW171 CW201 CW012 CW022 CW032 CW042
CW052 CW062 CW072 CW102 CW112 CW122
CW132 CW142 CW152 CW162 CW172 CW202
CW013 CW023 CW033 CW043 CW053 CW063
CW073 CW103 CW113 CW123 CW133 CW143
CW153 CW163 CW173 CW203 CW014 CW024
CW034 CW044 CW054 CW064 CW074 CW104
CW114 CW124 CW134 CW144 CW154 CW164
CW174 CW204 CW015 CW025 CW035 CW045
CW055 CW065 CW075 CW105 CW115 CW125
CW135 CW145 CW155 CW165 CW175 CW205
CW016 CW026 CW036 CW046 CW056 CW066
CW076 CW106 CW116 CW126 CW136 CW146
CW156 CW166 CW176 CW206 CW017 CW027
CW037 CW047 CW057 CW067 CW077 CW107
CW117 CW127 CW137 CW147 CW157 CW167
CW177 CW207 CW018 CW028 CW038 CW048
CW058 CW068 CW078 CW108 CW118 CW128
CW138 CW148 CW158 CW168 CW178 CW208
CW019 CW029 CW039 CW049 CW059 CW069

```
CW079 CW109 CW119 CW129 CW139 CW149
CW159 CW169 CW179 CW209 DMPR1 DMPR2
DMPR3 DMPR4 DMPR5 DMPR6 DMPR7 DMPR8
DMPR9 DMP10 VDR01 VDR02 VDR03 VDR04
VPR01 VPR02 VPR03 VPR04 VPR05 VPR06
VPR07 VPR08 VPR09 TAD01 TAD02 TAD03
TAD04 TAD05 TAD06 TAD07 TAD08 TAD09
TAD10 TAD11 TAD12 TAD13 TAD14 TAD15
TAD16 TAD17 TAD18 TAD19 TAD20 TAD21
TAD22 TAD23 TAD24 TAD25 TAD26 TAD27
TAD28 TAD29 TAD30 TAD31 TAD32 TAD33
TAD34 TAD35 TAD36 TAD37 TAD38 TAD39
TAD40 TAD41 TAD42 TAD43 TAD44 TAD45
TAD46 TAD47 TAD48 TAD49 TAD50 TAD51
TAD52 TAD53 TAD54 TAD55 TAD56 TAD57
TAD58 TAD59 TAD60 TAD61 TAD62 TAD63
TAD64 TAD65 TAD66 TAD67 TAD68 TAD69
TAD70 TAD71 TAD72 TAD73 TAD74 TAD75
TAD76 TAD77 TAD78 TAD79 TAD80 TAD81
TAD82 TAD83 TAD84 TAD85 TAD86 TAD87
TAD88 TAD89 TAD90 TAD91 TAD92 TAD93
TAD94 TAD95 TAD96 UDR02 UDR03 UDR04
UDR05 UDR06 UDR07 UDR08 UDR09 UDR10
UDR11 UDR12 UDR13 UDR14 UDR15 UDR16
500S1 500S2

XS=0
XN=32000
XD=110000-XN
)9MOVE XS XD XN
)9ASSM TERM,0,(SINTRAN)BPUN1-01:BPUN
XD<LLRTL
)BPUN ZRO

XD:056000      LLRTL:145236

)CLEAR
)9READ (SINTRAN)CX-BPUNSPM:BPUN
)9ASSM (SINTRAN)CX-ULISTSPM:SYMB
 **** 000000 DIAGNOSTICS ****

0/000000  OP2BG;OP2EN
)SYSDF
)9ASSM (AUX)LOIIST:SYMB
 **** 000000 DIAGNOSTICS ****

?
USIZE DSIZE UPLTS XSVTS XSVTS

)9ASSM TERM,0,(SINTRAN)BPUNSPM-01:BPUN
OP2BG<OP2EN
)BPUN ZRO

OP2BG:110000      OP2EN:173774

)CLEAR
0<-1
)ZERO
)9READ (SINTRAN)CX-BPUN4:SYMB
)9ASSM (SINTRAN)CX-ULIST4:SYMB
```

```
 **** 000000 DIAGNOSTICS ****

0/000000  BCSTA;OPEND
)SYSDF
)9ASSM (AUX)L01IST
 **** 000000 DIAGNOSTICS ****

?
CXCPU 3SINV GNFLA 5MAU1 5MAU1 5MAU1
5MAU1 JETTY JETTY WFRQI WFRQI AIRDW
  ACM CAMAC     GL  IOXN ASSIG TRACB
WFRQI   DIW  DOLW   US5   US6   US7
JETMO JETM2 P0EAR P1EAR JETM3 JETM4
JETM5 JETM6 T8INB MSYSU SVER3 ESCRE
RUBRE REMOT LOCAL STACO STOCO  LREM
CMSTA REPAS CLSTA LOCAL LEAV3 BUFIN


)9ASSM TERM,0,(SINTRAN)BPUN4-01:BPUN
BCSTA<OPEND
)BPUN ZRO

BCSTA:110000      OPEND:172205

)CLEAR
0<-1
)ZERO
0/000000  9SMRE;9EMRE
)9READ (AUX)BPUN-01-A:BPUN
)9ASSM (AUX)XSIULIST-01:SYMB
 **** 000000 DIAGNOSTICS ****

)SYSDF
)9ASSM (AUX)L01IST:SYMB
 **** 000000 DIAGNOSTICS ****

)9EXIT
@CONTINUE

)9ASSM 1,0,(AUX)BPUN-01-A:BPUN
9SMRE<9EMRE
)BPUN ZRO

)9EXIT
```

%% :: Part Two numeric Symbol List

SYMBOL LIST WITH ADDRESSES IN ASCENDING ORDER

```
SH2   =000000  SH430=000000  HH2   =000000  HH430=000000  CH2   =000000
CH430=000000  5BL01=000000  BUF   =000000  XHDNX=000000  XALOF=000000
89NMT=000000  99NFD=000000  ANLI  =000000  AKMCS=000000  SG02S=000000
SG10S=000000  SG11S=000000  SG17S=000000  SG20S=000000  SG21S=000000
SG27S=000000  SG37S=000000  SGLEN=000000  CSGLE=000000  XTCHN=000000
WFREE=000000  WLPAK=000000  WNBUF=000000  RRDR  =000000  LKEY  =000000
DLSTS=000000  FEOL  =000000  XCFUN=000000  FDPLX=000000  ERRNO=000000
MXX10=000000  MXX12=000000  8BDIS=000000  77XPO=000000  CCNO  =000001
ECBR7=000001  5BUFA=000001  MAXCA=000001  MAXUS=000001  XTSTA=000001
WNFRE=000001  WFPAK=000001  WNPAK=000001  WPCR  =000001  LBYTC=000001
CRET  =000001  DATAM=000001  HALF  =000001  ORERR=000001  CTCW  =000001
DH2   =000002  5ABLP=000002  MXLPR=000002  YIBNY=000002  BATNO=000002
5NBSG=000002  5ESSZ=000002  MATA1=000002  BINDX=000002  XTRTA=000002
WLLIM=000002  WPNUM=000002  WPOSS=000002  TXUND=000002  RRS   =000002
LMEM1=000002  LMEM  =000002  ADSTA=000002  XRESE=000002  MAMOD=000002
LHAST=000002  NTRMS=000003  NU5PR=000003  XTPRT=000003  WHLIM=000003
WPROC=000003  WDUM  =000003  WSAR  =000003  LMEM2=000003  CHEAD=000003
XHBYT=000003  XMLIS=000003  MCLEA=000003  IMODU=000003  SSTPC=000003
SPSGL=000004  XID13=000004  XIBNX=000004  5FDSZ=000004  NNBRT=000004
XNOBP=000004  XTMEM=000004  WBFSZ=000004  WTYP  =000004  WSCOM=000004
WCHL  =000004  LEOL  =000004  XHBUF=000004  INIT  =000004  IFSIZ=000004
MAXEM=000004  NOP   =000004  SLD1  =000005  NALME=000005  5SSSZ=000005
NOBGP=000005  5XLEV=000005  XLEVL=000005  XTMMX=000005  WBC   =000005
WSEQ  =000005  WTDR  =000005  PSTAT=000005  IRTRY=000005  HHMAX=000005
XTIME=000006  XTCMS=000006  WDEA  =000006  WPRI  =000006  WDADR=000006
RTSR  =000006  BCHEA=000006  DTLI  =000006  IDISP=000006  HHEAD=000006
XTSTS=000007  WSCA  =000007  WTCR  =000007  DRBI  =000007  XDASA=000010
NULDN=000010  8IOXT=000010  XTPRG=000010  BLDON=000010  RRTS  =000010
XBLDN=000010  DILEZ=000011  SYSST=000011  XTXRG=000011  WRTC  =000011
XID10=000012  XID11=000012  XID12=000012  XTTRG=000012  XTFRG=000012
RTTS  =000012  77MON=000013  9CXTI=000013  XTARG=000013  XTMRG=000013
WTTC  =000013  XTDRG=000014  RDMA  =000014  XTLRG=000015  ACCE5=000015
WDMA  =000015  HX21D=000015  XTSRG=000016  CPRIS=000016  .RDCR=000016
HX21S=000016  XTBRG=000017  WDCR  =000017  X2LMA=000017  XTAPR=000020
X22SI=000020  XTTAP=000021  MAX11=000021  77HOU=000022  XTUBF=000022
77MIN=000023  XTSBK=000023  XTSAD=000023  77DAY=000024  XTSBF=000024
XTCNT=000025  5TTSZ=000026  FRSSZ=000026  XTTCN=000026  8RTN  =000027
XTHOM=000027  XTASG=000030  XTRSG=000031  XTBMA=000031  XTBM0=000032
XTBM1=000033  NXRTP=000034  FMAX  =000034  5NP00=000034  XTBM2=000034
XTBMB=000034  XTBM3=000035  IDBUS=000040  MXLAM=000040  XLEV  =000040
DILGS=000044  8SGN  =000045  SPSG1=000045  COSEG=000046  F5DSG=000047
XLEVB=000050  L5DSG=000051  DSSNM=000052  SSSEG=000053  N5DSG=000053
SGNUM=000054  RTPNB=000063  RTDYN=000065  HDERC=000066  DSTAT=000067
DH430=000070  STPCN=000070  WTADR=000071  FDSTR=000072  SRRES=000072
SRDAT=000073  ACTSW=000074  MAINT=000075  HASTA=000076  5FYOP=000077
LISTP=000077  FLBPA=000100  SBMAX=000100  ENCLE=000100  LIINT=000100
CPCR  =000100  ENINI=000101  LISTF=000101  EUND  =000102  OMSG  =000102
ETOU1=000103  INTST=000103  ETOU2=000104  CMODI=000104  CWTTC=000104
CRTTC=000104  EILFN=000105  XRETR=000105  EILLI=000106  XREEN=000106
ETOMU=000107  XINIT=000107  ECLEA=000110  PCREG=000110  EINP  =000111
SAREG=000111  EPAR  =000112  CLENG=000112  EILSI=000113  DISP1=000113
ETOSM=000114  DISP2=000114  EILFZ=000115  MAXR  =000115  EX21  =000116
```

```
CHECK=000116  EDISP=000117  LISTL=000117  ELOCK=000120  DCBX =000120
MAX10=000120  5FCRL=000131  5FYER=000136  5FYFS=000144  S41FP=000146
MAX12=000151  MAX13=000151  LOADR=000200  DCEMS=000204  MOBU1=000344
MOBU2=000344  MOBU3=000344  MOBU4=000344  MOBU5=000344  MOBU6=000344
MOBU7=000344  MOBU8=000344  MOBU9=000344  MOB10=000344  MOB11=000344
MOB12=000344  MOB13=000344  MOB14=000344  MOB15=000344  MOB16=000344
CCL  =000347  CSAR =000377  5BLST=000400  BSF  =000403  NN5MS=000457
MSGMX=000474  5FYRL=000502  S504 =000504  S505 =000505  ULDN =000505
S506 =000506  OLDN =000506  S507 =000507  OFLDN=000507  S511 =000511
S512 =000512  S513 =000513  S514 =000514  MQLDS=000516  S516 =000516
5NASE=000517  S521 =000521  S522 =000522  S523 =000523  S524 =000524
S525 =000525  8RT2 =000531  5FYSP=000534  S554 =000554  S555 =000555
S556 =000556  S557 =000557  S561 =000561  S570 =000570  S571 =000571
S572 =000572  S573 =000573  CDVNR=000602  CADEV=000700  33CPU=000764
LBSEG=001000  ERB  =001000  5SDSP=001002  RDF  =001011  5DSK =001100
S1101=001101  S1102=001102  S1105=001105  S1106=001106  S1107=001107
S1110=001110  S1112=001112  S1114=001114  S1117=001117  S1120=001120
S1121=001121  S1122=001122  S1123=001123  S1124=001124  S1127=001127
S1130=001130  S1131=001131  S1132=001132  S1133=001133  S1134=001134
S1135=001135  S1142=001142  S1147=001147  S1150=001150  S1151=001151
S1152=001152  S1153=001153  S1154=001154  S1155=001155  S1160=001160
S1161=001161  S1162=001162  S1163=001163  S1164=001164  S1165=001165
S1166=001166  S1171=001171  S1172=001172  S1177=001177  SEMSE=001204
S1225=001225  S1226=001226  S1227=001227  S1230=001230  S1232=001232
S1233=001233  S1234=001234  S1235=001235  SGMAD=001266  S1300=001300
S1301=001301  S1333=001333  S1334=001334  S1335=001335  S1336=001336
S1337=001337  S1340=001340  S1341=001341  S1342=001342  WTM  =001414
ENSIZ=001520  VDD01=001716  VDD02=001717  VDD03=001720  VDD04=001721
S1732=001732  S1733=001733  S1734=001734  S1735=001735  S1737=001737
S1740=001740  S1741=001741  S1742=001742  S1743=001743  S1744=001744
S1745=001745  S1746=001746  S1747=001747  S1750=001750  S1751=001751
S1752=001752  DR1SI=002000  DR2SI=002000  FSERM=002003  DMS  =002004
MIBU1=002114  MIBU2=002114  MIBU3=002114  MIBU4=002114  MIBU5=002114
MIBU6=002114  MIBU7=002114  MIBU8=002114  MIBU9=002114  MIB10=002114
MIB11=002114  MIB12=002114  MIB13=002114  MIB14=002114  MIB15=002114
MIB16=002114  G5BUF=002256  GBSEG=002355  SYSSZ=002400  FSF  =002413
R3BUF=002736  R5BUF=002746  NLP  =003000  WRT  =003012  CLADB=003204
REW  =003404  77YEA=003700  BABST=003776  EMTY =004000  CLR  =004004
MABST=004131  FDABS=004215  BSB  =004403  RDB  =005001  ERG  =005414
SNS  =006004  .FSB =006413  LWR  =007002  RUN  =007404  33CPN=011610
BLOCK=016725  BUNLO=016725  CATES=016725  CARTT=016725  CBLOC=016725
CB2LO=016725  CBUNL=016725  SG05S=022000  5TTIF=022022  5PASS=022027
5SPAS=022030  5BCHF=022063  BSGM =026056  5FCOM=026055  5CSTC=027056
ENDOP=030330  BPOOL=030330  IOBUT=030331  EIOBU=030356  ZBDIS=030356
QDLAY=030361  9ESWP=031737  FPS41=031737  GSEG =032003  START=032017
SYSEV=032714  77CBU=033060  INTDF=033076  CONTX=033764  GNSTA=034000
NDEMF=034000  DEMFI=034030  CLFIE=034067  CLCFI=034074  OERRF=034103
IERRF=034117  DF1  =034147  TRT1 =034213  DF2  =034217  TRT2 =034263
DF3  =034267  TRT3 =034333  DF9  =034337  TRT9 =034403  DF11 =034407
TRT11=034453  DF13 =034457  TRT13=034523  DF20 =034527  TRT20=034573
DFRRT=034577  RRT1 =034643  SRRT1=034647  DISPE=034661  DT030=034721
DT037=034732  DT070=034743  DT075=034754  DT140=034765  DT135=034776
DT160=035007  DT288=035020  DT285=035031  DT300=035042  DT450=035053
DT460=035064  DTXXX=035075  DTYYY=035106  DTZZZ=035117  BIGDI=035222
DFDIL=035254  DALM1=035322  XP131=035327  XDILF=035331  DILBU=035334
XDILD=035352  MTFIE=035434  MTDI1=035537  MTDO1=035601  FDID1=036022
F1U01=036044  F1U00=036107  FPL3A=036153  LPL3A=036235  5TTST=036241
DT01R=036245  DT01W=036260  DT05R=036273  DT05W=036306  DT06R=036321
DT06W=036334  5TEND=036343  BD01R=036410  BD01W=036470  IDMO1=036555
UDMO1=036613  DMLP1=036677  DILP1=036764  MXLIN=037027  LINAR=037027
```

```
RPASS=037030  FRSEG=037042  ERTDT=037043  MPM4D=037046  SEMI1=037052
SEMI2=037056  SEMI3=037062  SEMI4=037066  SEMI5=037072  SEM60=037076
SEM61=037102  SEM62=037106  SEM63=037112  SEM64=037116  9SFIS=037122
REESM=037122  RTLFI=037126  FS4  =037132  FS5  =037136  FS6  =037142
FS7  =037146  FS20 =037152  FS21 =037156  FS27 =037162  FS61 =037166
F1101=037172  F1102=037176  F1136=037202  F1137=037206  F1142=037212
F1143=037216  F1144=037222  F1150=037226  F1151=037232  F1177=037236
D1201=037242  F1202=037246  F1204=037252  F1205=037256  F1222=037262
F1223=037266  F1731=037272  F2166=037276  F2167=037302  DDEBU=037306
UESEM=037332  NAMSE=037336  ACSEM=037342  SEMDL=037346  9EFIS=037352
F1200=037352  F1352=037356  BDSEM=037362  BASEM=037366  ID01I=037372
ID010=037422  IB01I=037453  IB010=037470  ID02I=037504  ID020=037534
F1203=037562  URERT=037566  SBANK=037600  DSBAN=037600  SBUF =037601
DFBLC=037602  FBLCK=037602  DTBLC=037604  TBLCK=037604  XFUNC=037606
NBLCK=037607  RPAR =037610  WPAR =037614  BT01R=037636  BT01W=037701
BT010=037726  BT01I=037742  CPRIM=037777  HDOF1=040003  HOLI1=040124
HDIF1=040145  HILI1=040275  HDOF2=040352  HOLI2=040473  HDIF2=040514
HILI2=040644  SEMPI=040710  PI001=040755  X21F1=041046  UDI01=041252
UDO01=041270  UDF01=041351  DTGP0=041412  OCTOP=041622  BRECC=041623
1NBPP=041624  2NBPP=041625  BNUMB=041626  BACKT=041627  BBCHT=041633
EBCHT=041634  BCHTA=041635  BNOBA=041642  DUMMY=041643  STSIN=041662
RTERR=041701  1SWAP=041720  RTDIL=041737  BPTMP=041756  RTSLI=041775
ACCRT=042014  TERMP=042033  5SWAP=042052  RWRT1=042071  RWRT2=042110
RWRT3=042127  RWRT9=042146  RTRFA=042165  DUMM2=042204  SPRT1=042223
COSP0=042242  WRT11=042261  WRT13=042300  WRT20=042317  TADAD=042336
9FPUD=042355  UDR01=042355  9LPUD=042374  BAK01=042374  BAK05=042413
BAK06=042432  2THSS=042451  BCH01=042451  9LBPR=042470  THISS=042470
TIMRT=042470  RTBES=042507  XSEGS=043240  XSGRT=043245  BPAG1=043246
SG2  =043252  SG3  =043257  SG4  =043264  SG5  =043271  SG6  =043276
SG7  =043303  SG10 =043310  SG11 =043315  SG12 =043322  SG13 =043327
SG14 =043334  SG15 =043341  SG16 =043346  SG17 =043353  SG20 =043360
SG21 =043365  SG22 =043372  SG23 =043377  SG24 =043404  SG25 =043411
SG26 =043416  SG27 =043423  SG30 =043430  SG31 =043435  SG32 =043442
SG33 =043447  SG34 =043454  SG35 =043461  SG36 =043466  SG37 =043473
SG40 =043500  SG41 =043505  S41FL=043511  SG42 =043512  SG43 =043517
SEGTX=043644  9ESGT=044135  TSLST=044135  TSLCO=044141  TSLNT=044145
S500S=044151  S500E=044265  X5   =044265  CCNOX=044333  CCSTA=044334
CCTAB=044335  IOXTA=044360  EIOXT=044370  S000 =044371  SDUMM=044372
S40D =044373  S41D =044376  S25D =044401  S32D =044407  SX7D =044412
S34D =044415  SX8D =044420  S563D=044427  S564D=044432  S33D =044435
SX1D =044443  SX2D =044446  SX3D =044451  SX4D =044454  SX5D =044457
SX6D =044462  D1232=044473  D1233=044476  D1234=044501  D1235=044504
D1225=044513  D1226=044516  D1227=044521  D1230=044524  SLDN =044533
WSLDN=044534  WULDN=044535  WFLDN=044536  FDFSE=044537  GLDN =044540
MOLDN=044541  BLDN =044542  DTHEA=044543  F0   =044543  DEVBU=044570
F1600=044570  ENDBU=044615  ENEN =044615  SPPR1=044616  SPTAB=044617
COSDA=044630  RFSGN=044642  ENDSP=044643  NRFSG=044643  RFSTB=044644
COSTA=044645  MLVSA=044645  MLVLO=044647  MLVUL=044662  TSTOW=044675
TSTDE=044677  TSTEN=044701  XRSTA=044710  BRSTA=044716  SETST=044752
T2XEX=045031  T1XEX=045033  T2BEX=045112  T1BEX=045114  STAX =045172
STTX =045175  STXB =045202  STTB =045205  STAB =045210  STADX=045267
STADB=045273  LDAX =045351  LDTX =045356  LDXB =045365  LDTB =045373
LDAB =045401  LTADX=045473  LTADB=045477  RSRES=045546  REDOM=045555
IEDCH=045575  BDOUT=045623  CBERS=045641  CXRES=045666  BDTOU=045715
CTIBA=045740  CTOBA=046057  BBREC=046271  BSTTY=046324  BTMOD=046377
CTMOD=046446  BCESC=046550  BSDAE=046645  BISIZ=046724  OISIZ=046726
PISIZ=047144  BOSIZ=047150  BSCPC=047160  ERRMO=047266  GTRT =047326
RTON =047335  RTOFF=047342  SLRMO=047353  FXTAD=047365  CHDFP=047402
PIOFL=047431  PIOFS=047435  SSCGE=047442  VSXGE=047455  GETBI=047472
PT3CO=047534  COPYB=047547  FLYTT=050052  XSETB=050073  SETBF=050116
```

```
XGTDF=050141  XSTDF=050160  SPTOW=050203  EDINB=050217  XCREA=050330
NONTT=050347  TTMWA=050350  ONTTM=050351  MBSPR=050352  MBPRV=050352
ASBPR=050353  AEBPR=050354  AEPRV=050355  APRVT=050356  TSLTU=050357
TSLSY=050360  TSLLO=050361  TSLHT=050362  TSLHA=050363  FPTSL=050364
DFPTS=050364  LPTSL=050365  TSLBR=050366  TSLES=050376  TSLLP=050406
TSLPR=050416  TSLTI=050456  TSLNE=050516  XSTBA=050556  TSTBA=050563
STBAC=050565  XS5ES=050610  S5ESC=050622  ANTIJ=050650  ANTI2=050657
RESAJ=050743  GTSLP=051067  MDSEG=051105  REFBP=051106  GBRKD=051116
CHLIM=051173  MMEXV=051265  ENTRC=051332  BLEVS=051370  GETDA=051476
LOFFA=051533  LONAL=051540  RTLMC=051561  TRTLD=051611  RTLRE=051645
MRTRG=051700  MRXRE=051701  MELRT=051702  MRCPL=051703  RTL2W=051726
RTLWF=051731  RTLRF=051734  ERIOT=052040  SET5N=052047  STOBV=052063
LOBYT=052067  STA6X=052073  LDT6X=052077  STAOX=052103  STA3X=052107
STA4X=052113  STA5X=052117  LDA1X=052123  LDA2X=052127  LDA3X=052133
STZOX=052137  STZ3X=052143  LDXOX=052147  MMLE =052153  TOFEN=052154
FILUS=052175  SVRBL=052220  RTCGP=052225  MLAMU=052243  CHARE=053167
GLCAC=053302  GAAC =053303  GLAPP=053311  GAPP =053312  CHLID=053320
SACLE=053330  CHLAM=053402  COLAM=053503  CLLAM=053537  LAMDI=053576
ELON =053606  ELOFF=053611  ELOFU=053614  DLOFU=053617  EUSEL=053622
DUSEL=053625  T2P07=053630  IBRSI=053645  T2C06=053650  T2P06=053651
CLRBM=053662  BMFRT=053676  BMTRT=053710  BMOR =053722  CSTSE=053756
REEIN=054002  REEUT=054004  MLOGI=054045  MLGTM=054050  MLIDF=054071
S3NOA=054240  SPTOP=054242  SPT3P=054244  GDRXM=054302  9GTLO=054411
GDBPA=054444  9ENTO=054463  CMBAB=054601  MBABP=054613  GBTIN=054700
PRLOG=054722  MOSTI=054740  MOTMI=055012  MOTRI=055021  MOSTO=055037
MOTRO=055115  MOTMO=055125  IOREM=055153  PLOTT=055163  PTBAS=055221
TBASE=055226  DAYEA=055226  DAMON=055232  ADDYE=055246  ADDMO=055247
PERIO=055250  HDAYE=055251  DAYES=055252  HOURS=055253  SECON=055254
HDAY2=055255  SEC2 =055256  PX9CL=055257  PX8CL=055260  X9CLO=055262
X9CL1=055263  X9CL2=055264  X9CL3=055265  X9CL4=055266  X9CL5=055267
X9CL6=055270  X9CL7=055271  PXCLX=055272  PPXCL=055272  PPX3C=055273
XCLNU=055274  XCLUN=055275  DXCLU=055276  PLXCL=055300  PWRFR=055302
ERUCL=055303  READC=055305  EX113=055322  EX112=055326  SETCL=055333
PANEL=055351  GERDV=055360  MGTMR=055401  MTCLD=055432  STMRS=055455
SMTBR=055466  MC144=055516  RTTAB=055524  DVTAB=055524  CBGET=055524
CBF1 =055653  CBF2 =055743  SG13S=056000  CBPUT=056026  MBFDI=056106
CAICL=056116  CAOCL=056155  SMTRA=056344  CMTRA=056466  SRETR=056512
CLPUT=056530  CLCLO=056652  TRDLP=057110  DMLPC=057113  DLPTM=057133
CHDVB=057152  DBSTW=057202  DBLDW=057204  PSTWO=057206  PLDWO=057212
IMTRI=057237  IMTRO=057353  MERRC=057446  M8OUT=057600  B8OUT=057602
T2P01=057652  T2P02=057655  MOURE=057700  M8INB=057723  B8INB=057725
B4INW=057727  T8INP=057731  HX21M=060000  T2P03=060006  T2P04=060013
T2P05=060020  GSGNO=060075  GRTDA=060101  GRTNA=060105  IPRIV=060121
XTLX =060162  STUSP=060316  STL12=060357  STL13=060375  LMASK=060377
HDTM2=060413  HDTM3=060416  X32ST=060437  X321T=060441  X321I=060443
O3CHA=060445  S3L13=060464  D3CHA=060473  X3T12=060515  X3T13=060517
INBX2=060540  EDTRM=060605  MPASE=060712  MPAGE=060714  RERRP=060770
NMAXU=061036  BUFFR=061037  CUIDX=061046  DBGPR=061047  BRPNT=061050
DEBUG=061055  ESCDB=061063  NACCO=061115  MRSEG=061175  RTDIR=061304
PRTDR=061337  3INST=061375  3OUTS=061557  CHERR=061642  C3OUT=061754
IOTR =061770  COPTD=062106  3OUTX=062137  3OWFI=062332  XPPRU=062343
USOX =062360  USO  =062360  US1X =062507  US1  =062507  US2X =062643
US2  =062643  US3X =062735  US3  =062735  US4X =063017  US4  =063017
BIOAC=063133  XWT05=063214  XRTDS=063242  XRTEN=063244  ZOPHY=063321
Z2PHY=063325  ZOUSR=063372  Z2USR=063376  ZXS12=063446  ZXS13=063454
MOCTB=063466  OCT13=063541  GETS2=063600  GET6 =063605  GET5 =063611
GET4 =063615  GET3 =063621  GET2 =063625  GET1 =063631  GET0 =063635
SWPCO=063667  SWPMA=063671  SWPMT=063673  SWPBL=063674  SWPPA=063675
PMTRA=063702  PPAGE=063707  P2PAG=063711  PPWFA=063713  PRESY=063715
PCOLD=063717  XTRAN=063736  DBTRA=064002  ATRAN=064017  MCALL=064052
```

```
MEXIT=064064  DECO =064147  OLEGS=064170  LEGSE=064171  R15ER=064205
RLEGS=064214  MOFIX=064257  MUNFI=064340  WSEG =064440  WSEGX=064524
REENT=064556  IMREE=064560  SPLRE=064645  CHCOR=064657  SSSWP=064661
CBREL=064716  CBRES=064731  LNK1S=064737  SWPRE=064777  MRLCL=065026
MLDIL=065042  ENTSG=065055  FXCTA=065205  PAGPN=065217  FIXC =065267
BFIXC=065420  FIXC5=065466  STMLE=065523  VTRNS=065531  TOPOF=065536
XTOPO=065555  SS41I=065572  SREEN=065606  TFWBA=065755  SGAND=066015
SGOR =066017  MXRET=066127  MXSIB=066153  SIBBD=066154  SIBAP=066157
DIA0 =066162  DIA1 =066211  DIA2 =066240  DSI0 =066267  DSI1 =066310
DSI2 =066331  EDSIB=066352  MAPSI=066352  MSIBB=066355  N500D=066417
S5CPU=066647  E5CPU=066647  5CPU1=066647  N500M=066700  CHIBA=067047
FSEMA=067470  XPION=067520  PCHSY=067560  XGETB=067707  CMOVB=067730
RELRE=070636  SVREI=071420  USVRE=071425  AATRA=071470  5ATRA=071472
5ALTO=071547  TS3CO=071572  SEGMO=071704  RSGMO=071735  5GETL=072016
5PUTL=072045  GETLS=072103  5GSYD=072201  5GNSE=072211  RESNA=072340
RELNA=072417  ALLRE=072454  CH5MX=072651  5BABO=072704  1RL5P=072741
RL5PD=072742  FROME=073043  ESC50=073456  5ESCR=073456  XONES=073551
XOFFE=073562  ESC5O=073575  PIONP=073710  S5TSL=073737  5S1TS=073763
5R1TS=073765  N500S=074035  5SWRT=074111  N500T=074221  LOWAC=074236
RIOWA=074255  5DTUP=074270  T5RST=074356  IBMOV=074422  5BPUT=074670
PNW5S=074721  PT5RS=074741  X5DCN=074750  CHEXQ=075010  APLRS=075050
X5MCS=075062  5MCST=075063  6FXBN=075203  5FXTB=075204  5DSPS=075205
5MBBA=075206  CERNE=075207  5DBFL=075210  MUDMA=075211  UDMA =075244
UWT11=075260  UDTMO=075302  PIOCM=075313  PDRIV=075320  PITIM=075323
PIORE=075336  7ENDC=075367  98END=075367  N5ADR=100000  SILFO=100000
OVERR=100000  INTCH=102164  9POFS=106000  9SMRE=106000  BRK0 =106000
BRK1 =106010  BRK2 =106020  BRK3 =106030  BRK4 =106040  BRK5 =106050
BRK6 =106060  ECH0 =106070  ECH1 =106100  ECH2 =106110  ECH3 =106120
ECH4 =106130  ECH5 =106140  ECH6 =106150  12DFO=106160  10DFO=106161
IISTA=106162  OISTA=106163  STTIN=106165  TYENT=106170  ECHSU=106533
ECAPD=106612  TECHO=106644  TBREA=106713  TDGET=107025  TTGET=107027
GECHO=107157  XOFTR=107203  10BUF=107323  DWRIT=107327  SG03S=110000
SG04S=110000  SG06S=110000  SG07S=110000  SG14S=110000  SG15S=110000
SG22S=110000  SG23S=110000  SG26S=110000  SG32S=110000  SG36S=110000
SG42S=110000  SG43S=110000  5LREG=110000  5AREG=110000  BADM =110001
SGBRK=110001  PTSIN=110003  NAMTA=110003  TSLAN=110051  SRTER=110052
ENDNT=110055  FFTAB=110055  DIRTA=110060  STAGP=110062  STOGP=110074
ENDHD=110107  XSBPR=110111  ENDDD=110136  IOAPD=110156  ENDDT=110165
BFBUF=110165  SRTON=110177  SMEND=110205  SMSTR=110232  ENDBF=110233
FCPUN=110233  MCCLC=110234  OOAPD=110243  SMENT=110250  XOOAP=110252
CXRBP=110264  SM3LE=110273  SMXLE=110306  CXRBG=110313  SM2LE=110321
SMLEA=110332  RBGET=110344  RBPUT=110364  SMCHT=110373  TDBPU=110410
DBPUT=110412  SMYES=110433  SMABL=110451  TDBGE=110456  DBGET=110460
RWGET=110516  RWPUT=110534  SM1AB=110544  TELIN=110561  TELEN=110564
SMGCO=110656  CHDL1=110725  CHDL2=110775  SMEDI=111020  XONCH=111027
OXONC=111120  XONRE=111144  XONWR=111177  M8OTE=111232  B8OTE=111234
M8OIN=111316  B8OIN=111332  M8INT=111353  M8IBT=111466  VDUST=111575
VDUSB=111607  GETSO=111617  IB8IN=111620  B8INT=111624  SRCSH=111631
SMSRC=111657  SMTMT=111720  SMTIM=112007  IB4IN=112043  B4INT=112045
SAGPA=112110  SMSGP=112115  SMAGP=112122  SMGPA=112130  P3OST=112267
PONIO=112572  XPFAD=112771  XMSGA=112772  ZXCOM=113005  CLXMS=113015
PFXMS=113032  MFXMS=113041  ZXRES=113050  ZXRRS=113057  SMKGP=113061
ZXTRS=113066  P2XMS=113075  XDHOM=113106  DRXMS=113112  SMCRE=113120
SMORE=113125  SMCWR=113133  SMOWR=113140  XMSGY=113165  SMBAC=113167
SMSCA=113174  SMCCL=113216  SMOUT=113230  ZBINI=113252  ZBGET=113311
SMDDE=113317  SM2DE=113401  ZBREL=113401  SMDOC=113442  PTABL=113463
SMOCT=113473  LPPUT=113503  SM3OC=113514  PCON1=113516  SMDTD=113534
SMDEC=113540  DPRIN=113540  SMPER=113600  RPIT3=113605  SPT3W=113612
SEPTS=113631  CTRDI=113632  SMSPA=113663  SMCRL=113675  SM2TC=113704
SMTCO=113715  SMTCI=113723  SMWIN=113735  MCSTA=114000  XSPT3=114000
```

```
PFIXC=114004  SMDYN=114012  SMTAC=114037  SMFIB=114051  XRELE=114106
MBSRE=114124  PTERM=114136  6PINO=114150  6BFTY=114162  6DFTY=114165
6PRTN=114170  6TILB=114176  6PARE=114211  6PSEG=114235  BUSYE=114251
6PLOG=114256  6PADD=114267  6PSPN=114275  6FFHE=114307  MDRIV=114307
PFEIL=114312  6FFTR=114322  DRFEI=114322  6TILL=114334  6P3RE=114345
RETUA=114345  RETSG=114346  6PRTF=114347  6RTLR=114352  6CTAB=114360
6TINC=114361  6BLCK=114362  6BLOC=114363  6CFIL=114364  6FINA=114365
NWORD=114366  6RPAR=114367  6WPAR=114374  6IDMA=114401  6IDTA=114405
6EIDT=114411  6SINT=114415  COOPT=114420  6MACM=114424  6TERR=114434
SMCIL=114434  6ILLA=114446  6STRN=114456  6ADRM=114463  6XALT=114464
6CFUP=114465  SMFAM=114474  SMTXF=114506  SMCAM=114517  SCOMT=114532
UPDIE=114552  DIDLO=114611  DEDF1=114635  WRDIL=114715  DVDIL=114764
5SGF1=115106  SGFIX=115110  CCHD1=115234  TSTSH=115364  DTAPR=115533
GETDW=115575  PUTDW=115634  BUSCD=115705  ENT13=116025  IRTCP=116076
CMCLG=116077  MCLGB=116100  TNMCA=116101  MCLGF=116102  MCLGP=116103
MCLGO=116104  MGOTA=116105  FPFCO=116705  OUTLI=116735  POFMO=116763
XLINK=117015  POFNM=117030  SINSE=117036  SSMON=117057  SSSLE=117153
SSLEA=117154  3MAPS=117216  6ERRP=117221  6ERRO=117223  SMFIL=117235
6OOPE=117240  NW9ER=117460  3MSIB=117472  DBUGS=117537  DATSE=117540
DBGSE=117541  OLDSE=117542  BPHAN=117543  REGBL=117544  APTNO=117545
3BRPN=117546  DRTDE=117562  3DEBU=117645  COLDS=117736  RESYS=117741
DSTEN=120006  CSWLG=120134  CCSWL=120135  SWLGO=120136  TFPF1=120137
TFPFS=120137  TFPF2=120140  CFPF1=120141  CFPFS=120141  RIMAG=120142
CFPF2=120142  TPFR1=120143  TPFRS=120143  TPFR2=120144  CPFR1=120145
CPFRS=120145  CPFR2=120146  TPF41=120147  TPFL4=120147  TPF42=120150
CPF41=120151  CPFL4=120151  CPF42=120152  TPF11=120153  TPFL1=120153
TPF12=120154  CPF11=120155  CPFL1=120155  CPF12=120156  SNWP1=120157
SNWPS=120157  SNWP2=120160  FIXCR=120161  RMEMO=120162  SEGAD=120164
RSAVE=120202  SELAR=120237  FINDI=120245  PAGE2=120324  PAGEF=120327
ASKAR=120365  3MLOG=120371  LEGSC=120411  CHRTD=120434  LIMCH=120444
IMTAB=120456  IITAB=120504  ISTAB=120533  SLAMU=120537  CLAMU=120541
FFMTA=120565  FFITA=120573  3GSGN=120600  FFSTA=120601  3GRTD=120602
ITIMT=120624  RTIMT=120630  CLWIN=120656  CLSEG=120661  STSEG=120672
SCPOU=120702  CLPAG=120706  CLERT=120723  CERR1=120737  SETRT=120751
6SOPE=120772  6MOPE=120776  SEGCH=121004  6CLOS=121015  3GRTN=121017
CLNRE=121050  INORO=121055  CSEGS=121066  XCSEG=121077  R1IMA=121102
R1MEM=121110  W1IMA=121114  W1MEM=121122  R1SAV=121126  W1SAV=121161
CHREE=121164  WXSAV=121201  COUTL=121213  WIMBA=121214  IIDTA=121251
RIDTA=121253  IEIDT=121255  REIDT=121257  INSRP=121332  3SPLR=121335
SEGIN=121416  ADUMP=121626  ODUMP=121630  STSLI=121662  LRU   =121665
SG30S=122000  SG31S=122000  FPT2E=122003  DBASE=122005  KEXMC=122006
GETLA=122044  T3REP=122103  SEGSO=122120  IBACK=122200  RBACK=122202
RANDO=122203  GDEV3=122265  XTRNS=122276  TRNSE=122305  RWSEG=122373
3GETX=122427  DUSSU=122536  DURSU=122603  3EXAB=122632  DURPR=122655
LINKO=122661  SREMO=122777  DVSTR=123011  DFPRM=123016  SRESE=123023
ELON3=123042  T207R=123074  ZSREL=123110  SRELE=123113  ELOF3=123116
ELO3F=123157  DLOF3=123161  ATRNS=123200  EUSE3=123226  DUSE3=123230
OVERL=123247  IBR3S=123277  9FINF=123314  9CABL=123314  9CLOG=123315
9CUNI=123316  1CFST=123317  9CFST=123317  2CFST=123320  1CFSI=123321
9CFSI=123321  2CFSI=123322  9CCDF=123323  9CFNO=123324  SMGFI=123327
SHRSO=123365  PCCS =123436  PIERE=123456  PIDRE=123457  T206R=123466
PWFAI=123475  PT3MC=123515  RESTA=123554  SETPT=123631  GMAIN=124023
DLSGF=124143  DEFSG=124145  TBUSP=124444  RSPTA=124537  LEV1 =124622
CALAB=124666  LEV14=124772  XCCTA=125002  RMAGT=125016  LAMAR=125024
NSWPA=125074  NINIT=125145  CUMTA=125215  USIOX=125241  SPCOR=125252
CRBIN=125316  SPARM=125357  PT3FU=125420  IIIOX=125604  RFIOX=125606
LISCO=125714  X2A00=125715  X2A02=125735  X2A05=125755  X2A10=125775
X2A11=126015  TUSON=126026  X2A12=126035  X2AST=126055  X2ATL=126075
X2LL =126115  X2LL1=126116  X2LL2=126117  X2LL4=126120  X2LL5=126121
STORE=126122  X2NEW=126270  X20LD=126271  X212S=126275  X213S=126306
```

```
X2SND=126372  X21PG=126414  CHVAR=126417  GOODR=126424  X2SSS=126437
X2STA=126502  X21ER=126542  X21UT=126554  X2SBY=126570  X2RAC=126572
X2RBY=126574  XTSTC=126576  SWDEN=126611  X21AA=126620  MMBRE=126647
5MBRE=126653  ILUTA=126737  RLUTA=126741  X21RE=127003  X21CL=127054
DUMCA=127113  X21CN=127130  X21RD=127254  X21DC=127256  X21CH=127303
SSDNU=127331  X21BR=127346  X21C =127360  X21LV=127433  X21IN=127442
DBSPR=127445  LIADD=127575  X21GC=127633  ISPHE=127635  RSPHE=127637
X21TO=127713  X2GET=130027  RHIFE=130042  500HI=130052  SCHPR=130061
X211 =130072  X2118=130177  X2114=130220  X21SH=130251  X2119=130277
X2120=130306  X2121=130326  X2116=130355  X212 =130401  GMUSI=130506
X217 =130560  X2110=130667  SCE06=130731  X2112=130733  X21RT=130743
X218 =130770  X219 =131012  X2S01=131032  GUSEN=131070  5PF1 =131125
5PF2 =131130  5PF3 =131133  X2E01=131232  PT3UD=131232  CDAFI=131361
DECOM=131403  DECOE=131462  TRAPD=131526  ROBJE=131533  DTTU =131710
WOBJE=131711  DTFU =131712  URT01=131740  DISPR=132076  ERRF =132150
DILRT=132155  N500C=132216  CDRTC=132326  PT3MB=132354  ISYSE=132537
SCPRO=132620  SPORT=132677  SCSFS=132770  XITO5=133060  9BPTM=133063
ITO50=133065  GFILI=133075  IFM50=133206  FR5TM=133236  IACT5=133247
TER50=133365  CHIOB=133404  DIRUN=133414  RRTWT=133434  SBPRT=133436
ACTRT=133451  ACTXR=133504  EBPRT=133510  PRVTT=133510  5ACTS=133527
EPRVT=133550  DEMSV=133550  APIOC=133551  PIRET=133611  HRSTA=133620
RSTAR=133626  EXEL =133642  GFIAC=133672  XINST=133771  PM116=133775
XMREM=134032  PM160=134032  ND852=134042  UDFUN=134054  DUMF =134062
WAIF =134065  XMCHK=134100  COAF =134135  REMCM=134162  MCHAN=134252
CCCOM=134274  SENCT=134403  CENCT=134405  UASTR=134622  UADDR=134655
UBUSY=134660  UFIN =134663  SGWPR=134664  SGWPE=134670  MDEAB=134707
UD11T=134714  PREAD=134751  PWRIT=134761  KREAD=134764  OLDL =134774
OLDH =134775  PICAC=134777  FIXOK=135022  DFHDL=135072  LIHDL=135076
PISEF=135103  CHPO =135164  CHRUN=135212  PIOCE=135232  PILOA=135240
PISTO=135273  LPOIN=135312  LKICK=135313  PISTA=135316  STMSY=135444
FNO  =135445  MESMA=135446  ORBIT=135451  PIRES=135540  PIREL=136570
PIKIC=135612  PIWKI=135661  GTDF =135710  PIUNL=135715  DVMAS=136007
FRUSE=136051  DATIM=136052  BITTA=136054  MCCLD=136253  MESBU=136454
APDRI=136571  XEPT3=136672  SETCP=136674  XSETC=136675  NW5ST=136707
SPITM=137017  EMONI=137042  OKMON=137046  MLCTA=137051  MONIC=137051
MCCO =137065  HISTS=137111  CLEST=137164  SMMAI=137165  OFFES=137243
ONESC=137263  CH5CP=137315  MESSR=137352  MSINI=137716  YOUHA=137735
MLHEL=137771  XLPT3=137777  SG12S=140000  SG24S=140000  SG25S=140000
SPRIM=140000  FSSTA=140000  LIDTE=140037  MLINI=140042  INMES=140102
MODIN=140111  MODI1=140112  MLSTO=140122  MLRUN=140125  HIDTE=140136
MLSEN=140176  MLBRO=140200  EXPAN=140404  MLDSE=140455  MLDBR=140634
MODUT=140760  MLLME=141120  MLLBR=141122  CTRMA=141265  CTR2M=141265
MLDLM=141346  MLDLB=141350  READM=141576  WMESS=141672  RBUFF=141763
WBUFF=141770  BTSTO=142011  BTCLE=142032  XBTLO=142054  BTLOA=142074
MFIN =142100  XFIN =142122  5BUFR=142134  GOMAG=142226  SSBUF=142234
MAGTR=142277  MBUSY=142312  XMBUS=142332  LAMFU=142334  PAR1 =142335
MFINI=142336  PAR2 =142336  PAR3 =142337  LAPLI=142340  PARA1=142341
PARA2=142342  PARA3=142343  NSLAR=142344  LWIAR=142355  MFEIL=142361
NFLAD=142366  TPPAG=142405  TNPAG=142421  TLID =142433  TRFP =142441
PTLAM=142532  MFRES=142606  SFTIM=142634  SFFUN=142736  TFDIS=142753
PFLAM=143203  FINFD=143276  CRLAM=143452  DELAM=143541  LAPRO=143612
CFDIS=143613  FLVIS=143617  FDIBU=143627  BUFDI=143641  FDIFE=143643
FDIFI=143663  TLPRI=143700  CLP10=143756  XTRDL=144001  LAREA=144010
LAINF=144156  CONVC=144240  M8IDR=144302  M88IW=144316  B8IWO=144417
LLAMC=144461  NN310=144514  SLAMC=144560  SGPIB=144614  B8IIN=144622
M8IIN=144622  B4IIN=144727  HDLC =145020  IMHDL=145162  PZCRA=145240
ACT12=145246  ACT13=145257  KGPIB=145261  ICHAI=145276  SGPBS=145304
OCHAI=145305  TSTCH=145355  DICHA=145361  DOCHA=145374  GVAL =145405
ZCRAS=145413  PRTCG=145436  GEPNE=145456  GPIIM=145477  ACTAB=145507
SMOOP=145522  FUNCT=145551  IOACT=145614  COMRT=145762  ALLRT=145763
```

```
LGARR=145764  LOG1 =146031  BADDV=146152  PGARR=146331  BSPAC=146467
MSGLE=146504  PGAEN=146507  SLVTB=146507  LOTH1=146533  LOTH2=146534
HISTO=146535  GSYS =146574  GUSE =146667  FDVBU=146733  HTEST=146743
HHIST=146744  HFLGH=146751  HISTI=146753  XHIST=147030  DDRTB=147045
SHIST=147061  TAPEF=147067  MDRTB=147133  HDSTA=147144  FALSE=147145
FBFBU=147171  BACKX=147205  XMPAT=147220  SPOOD=147237  LTOUT=147245
SBYTC=147254  SADTS=147260  SCRET=147263  SPOOS=147265  RACTB=147266
RBYTC=147271  DRERR=147274  GPUPI=147303  ZXCHK=147316  XSDAT=147336
SPBUF=147417  RFTS =147431  GDEBU=147507  HOINT=147525  PRSDC=147565
POFTO=147606  XSSND=147620  HDREC=147641  ACHSU=147742  STWOR=147747
CCCIN=147756  CINBT=147765  SG16S=150000  SNETS=150000  ENETS=150003
M5SGS=150032  HIINT=150073  RCALA=150114  RFDSG=150142  LDVFC=150240
SDATA=150247  DATA =150257  FENTR=150275  HXST =150301  XSHDR=150316
SMCLE=150327  FLEAV=150340  ISTCK=150364  WBACK=150407  HDSIN=150414
MINBT=150442  MOUTB=150456  M5CPU=150472  TRASE=150564  RECSE=150611
ACE91=150645  SPSTA=150672  9BPEX=150702  MPRVT=150732  LOOKA=150736
3SETC=150757  3RCLO=151162  GPIBT=151507  MBTER=151530  TERMS=151675
GPBUS=152124  GPFIN=152135  CCTCO=152237  GPERR=152461  LISTR=152602
EGXMS=153000  SCALL=153026  TREMO=153072  GTSUB=153112  GPIBH=153177
NFERR=153201  XRERR=153207  XSERR=153213  XERR =153214  FATER=153217
GPIBO=153301  SMFFL=153330  UELGN=153454  STFLG=153562  SMTRS=153672
MACD =153777  OCTOB=154312  STABA=154424  STOBA=154426  XBADM=154430
SRMMO=154500  SLCMO=154502  TOOPC=154511  STARC=155616  RKTAS=155623
DTSLI=155643  5ERET=155735  DRXAC=155750  WRMHE=155773  CREME=156030
CRHEE=156106  CRHEO=156167  DEFEL=156171  GETME=156254  GETVA=156304
SAVE =156332  CHKRN=156347  UPDMB=156347  CHCLA=156373  GETMB=156421
BYTPU=156443  WORDP=156530  BYTGE=156624  LOADB=156672  CMLTS=156713
STORB=156727  BDPUT=156766  BDGET=157064  CPTSL=157150  BDTCH=157172
BM8OU=157212  BB8OU=157214  BB4IN=157345  LTSPR=157404  BB8IN=157541
CNVER=157742  PUTPO=157747  GETPO=157767  MOVIT=160024  CLIDA=160050
CGTTS=160117  ITSLI=160155  XRTSL=160157  CLODA=160214  SNDBU=160310
SNDRF=160331  SNDRE=160455  CBREC=160635  BDECH=160700  BDBRE=161016
NWSTA=161171  SNDCP=161302  CTRME=161374  BDTMO=161471  BDTTY=161551
BDDES=161601  BDDUM=161632  HIGHT=161645  RDATR=161646  EBUFF=161647
BDESC=161650  RLOCA=161651  BDDIS=161652  CESCR=161653  RESCF=161654
NWREM=161655  ISZRS=161656  ERRSP=161657  TREPS=161660  DMMES=161661
RFIMS=161664  ERESP=161667  EDRSP=161672  INIBD=161675  INISN=161756
LOGDI=162027  BDRIN=162035  ENSYS=162122  BERES=163022  STOTA=163042
BDLOU=163066  BDDSC=163132  BDROU=163171  FAERO=163232  FAERR=163234
BDRWT=163236  SETSI=163253  CHPAR=163312  CHSIZ=163345  REJEC=163400
MXMSG=163517  YENDC=163535  DV000=163535  EN000=163652  DV200=163652
EN200=163767  DV300=163767  SG40S=164000  EN300=164170  DV400=164170
EN400=164171  DV500=164171  EN500=164336  DV600=164336  EN600=164343
DV700=164343  EN700=164344  D1000=164344  E1000=164347  D1100=164347
E1100=164550  D1200=164550  E1200=164745  D1300=164745  CMMCL=164746
E1300=165146  D1400=165146  E1400=165151  D1500=165151  9EBAD=165152
E1500=165152  D1600=165152  E1600=165155  D1700=165155  E1700=165304
D2000=165304  E2000=165305  D2100=165305  E2100=165466  D2200=165466
E2200=165473  D2300=165473  E2300=165514  D2400=165514  E2400=165515
TMRTA=165515  M1TMR=165516  9SXTD=165544  TMRTE=165546  9EXTD=165554
ETMRT=165556  ITB10=165557  ITB11=165677  ITB12=165720  SWPLO=166062
ITE10=166071  ITE11=166104  ITE12=166123  ITE13=166140  IT13E=166145
ITB13=166145  ID10T=166316  ID12T=166316  LP1BU=166316  XBCKT=166516
BFDIS=166524  BDISK=167045  FCPUL=167545  CMCPU=167577  CMCHT=170520
DEVTI=171274  XSTAT=171275  DENSI=171276  FUNCO=171277  UNINO=171300
COCOD=171301  SMAGT=171307  ERRET=171543  ERREX=171557  BURET=171574
FINET=171613  RELOA=171626  FUNIL=171642  COMEA=171657  ALOAF=171726
ADR2B=171730  LDRAD=171733  XSWTP=171734  COMEB=171757  999EN=171764
TIOBU=171764  TADFP=172000  TADLP=172001  TDLPH=172002  FMEMC=172003
COMEC=172036  KLRC1=172117  COMED=172122  SBFIE=172204  LDEND=172205
```

```
OPEND=172205  COMEX=172213  STARE=172237  SINTR=172243  BAUTI=172337
STAKO=172345  STACH=172432  77EMR=172727  9EMRE=172727  XCORM=172727
9EIOB=173417  SSCLD=173652  RSCLD=173730  OP2EN=173774  XCHIO=174215
YGETA=174421  XGETA=174423  XINIP=174706  XMEMC=174747  XFPGN=174751
XPHYS=174763  IOUTL=175040  XOUTL=175042  ZEROP=175134  PSTWR=175155
MAKET=175160  99END=175676  SPRIS=177762  HXDOK=177767  HXTMO=177770
TERMC=177777  WPVER=177777  XXRPT=177777  ILL  =177777
```

%% :: Part Two alphabetic Symbol List

SYMBOL LIST IN ALPHABETIC ORDER

```
,FSB =006413  ,RDCR=000016

0LEGS=064170

10BUF=107323  10DFO=106161  12DFO=106160  1CFSI=123321  1CFST=123317
1NBPP=041624  1RL5P=072741  1SWAP=041720

2CFSI=123322  2CFST=123320  2NBPP=041625  2THSS=042451

33CPN=011610  33CPU=000764  3BRPN=117546  3DEBU=117645  3EXAB=122632
3GETX=122427  3GRTD=120602  3GRTN=121017  3GSGN=120600  3INST=061375
3MAPS=117216  3MLOG=120371  3MSIB=117472  3OUTS=061557  3OUTX=062137
3OWFI=062332  3RCLO=151162  3SETC=150757  3SPLR=121335

500HI=130052  5ABLP=000002  5ACTS=133527  5ALTO=071547  5AREG=110000
5ATRA=071472  5BABO=072704  5BCHF=022063  5BL01=000000  5BLST=000400
5BPUT=074670  5BUFA=000001  5BUFR=142134  5CPU1=066647  5CSTC=027056
5DBFL=075210  5DSK  =001100  5DSPS=075205  5DTUP=074270  5ERET=155735
5ESCR=073456  5ESSZ=000002  5FCOM=026055  5FCRL=000131  5FDSZ=000004
5FXBN=075203  5FXTB=075204  5FYER=000136  5FYFS=000144  5FYOP=000077
5FYRL=000502  5FYSP=000534  5GETL=072016  5GNSE=072211  5GSYD=072201
5LREG=110000  5MBBA=075206  5MBRE=126653  5MCST=075063  5NASE=000517
5NBSG=000002  5NPOO=000034  5PASS=022027  5PF1  =131125  5PF2  =131130
5PF3  =131133  5PUTL=072045  5R1TS=073765  5S1TS=073763  5SDSP=001002
5SGFI=115106  5SPAS=022030  5SSSZ=000005  5SWAP=042052  5SWRT=074111
5TEND=036343  5TTIF=022022  5TTST=036241  5TTSZ=000026  5XLEV=000005

6ADRM=114463  6BFTY=114162  6BLCK=114362  6BLOC=114363  6CFIL=114364
6CFUP=114465  6CLOS=121015  6CTAB=114360  6DFTY=114165  6EIDT=114411
6ERRO=117223  6ERRP=117221  6FFHE=114307  6FFTR=114322  6FINA=114365
6IDMA=114401  6IDTA=114405  6ILLA=114446  6MACM=114424  6MOPE=120776
6OOPE=117240  6P3RE=114345  6PADD=114267  6PARE=114211  6PINO=114150
6PLOG=114256  6PRTF=114347  6PRTN=114170  6PSEG=114235  6PSPN=114275
6RPAR=114367  6RTLR=114352  6SINT=114415  6SOPE=120772  6STRN=114456
61ERR=114434  6TILB=114176  6TILL=114334  6TINC=114361  6WPAR=114374
6XALT=114464

77CBU=033060  77DAY=000024  77EMR=172727  77HOU=000022  77MIN=000023
77MON=000013  77XPO=000000  77YEA=003700  7ENDC=075367

89NMT=000000  8BDIS=000000  8IOXT=000010  8NOBA=041642  8RT2  =000531
```

```
8RTN =000027  8SGN =000045

98END=075367  999EN=171764  99END=175676  99NFD=000000  9BPEX=150702
9BPTM=133063  9CABL=123314  9CCDF=123323  9CFNO=123324  9CFSI=123321
9CFST=123317  9CLOG=123315  9CUNI=123316  9CXTI=000013  9EBAD=165152
9EFIS=037352  9EIOB=173417  9EMRE=172727  9ENTO=054463  9EPT3=113631
9ESGT=044135  9ESWP=031737  9EXTD=165554  9FINF=123314  9FPUD=042355
9GTLO=054411  9LBPR=042470  9LPUD=042374  9POFS=106000  9SFIS=037122
9SMRE=106000  9SXTD=165544

AATRA=071470  ACCE5=000015  ACCRT=042014  ACE9I=150645  ACHSU=147742
ACSEM=037342  ACT12=145246  ACT13=145257  ACTAB=145507  ACTRT=133451
ACTSW=000074  ACTXR=133504  ADDMO=055247  ADDVE=055246  ADR2B=171730
ADSTA=000002  ADUMP=121626  AEBPR=050354  AEPRV=050355  AKMCS=000000
ALLRE=072454  ALLRT=145763  ALOAF=171726  ANLI =000000  ANTI2=050657
ANTIJ=050650  APDRI=136571  APIOC=133551  APLRS=075050  APRVT=050356
APTNO=117545  ASBPR=050353  ASKAR=120365  ATRAN=064017  ATRNS=123200


B4IIN=144727  B4INT=112045  B4INW=057727  B8IIN=144622  B8INB=057725
B8INT=111624  B8IWO=144417  B8OIN=111332  B8OTE=111234  B8OUT=057602
BABST=003776  BACKT=041627  BACKX=147205  BADDV=146152  BADM =110001
BAK01=042374  BAK05=042413  BAK06=042432  BASEM=037366  BATNO=000002
BAUTI=172337  BB4IN=157345  BB8IN=157541  BB8OU=157214  BBCHT=041633
BBREC=046271  BCESC=046550  BCH01=042451  BCHEA=000006  BCHTA=041635
BD01R=036410  BD01W=036470  BDBRE=161016  BDDES=161601  BDDIS=161652
BDDSC=163132  BDDUM=161632  BDECH=160700  BDESC=161650  BDGET=157064
BDISK=167045  BDLOU=163066  BDOUT=045623  BDPUT=156766  BDRIN=162035
BDROU=163171  BDRWT=163236  BDSEM=037362  BDTCH=157172  BDTMO=161471
BDTOU=045715  BDTTY=161551  BERES=163022  BFBUF=110165  BFDIS=166524
BFIXC=065420  BIGDI=035222  BINDX=000002  BIOAC=063133  BISIZ=046724
BITTA=136054  BLDN =044542  BLDON=000010  BLEVS=051370  BLOCK=016725
BMBOU=157212  BMFRT=053676  BMOR =053722  BMTRT=053710  BNUMB=041626
BOSIZ=047150  BPAG1=043246  BPHAN=117543  BPOOL=030330  BPTMP=041756
BRECC=041623  BRK0 =106000  BRK1 =106010  BRK2 =106020  BRK3 =106030
BRK4 =106040  BRK5 =106050  BRK6 =106060  BRPNT=061050  BRSTA=044716
BSB  =004403  BSCPC=047160  BSDAE=046645  BSF  =000403  BSGM =026055
BSPAC=146467  BSTTY=046324  BT01I=037742  BT01O=037726  BT01R=037636
BT01W=037701  BTCLE=142032  BTLOA=142074  BTMOD=046377  BTSTO=142011
BUF  =000000  BUFDI=143641  BUFFR=061037  BUNLO=016725  BURET=171574
BUSCD=115705  BUSYE=114251  BYTGE=156624  BYTPU=156443

C30U1=061754  CADEV=000700  CAICL=056116  CALAB=124666  CAOCL=056155
CARTT=016725  CATES=016725  CB2LO=016725  CBERS=045641  CBF1 =055653
CBF2 =055743  CBGET=055524  CBLOC=016725  CBPUT=056026  CBREC=160635
CBREL=064716  CBRES=064731  CBUNL=016725  CCCIN=147756  CCCOM=134274
CCHDI=115234  CCL  =000347  CCNO =000001  CCNOX=044333  CCSTA=044334
CCSWL=120135  CCTAB=044335  CCTCO=152237  CDAFI=131361  CDRTC=132326
CDVNR=000602  CENCT=134405  CERNE=075207  CERR1=120737  CESCR=161653
CFDIS=143613  CFPF1=120141  CFPF2=120142  CFPFS=120141  CGTTS=160117
CH2  =000000  CH430=000000  CH5CP=137315  CH5MX=072651  CHARE=053167
CHCLA=156373  CHCOR=064657  CHDFP=047402  CHDL1=110725  CHDL2=110775
CHDVB=057152  CHEAD=000003  CHECK=000116  CHERR=061642  CHEXQ=075010
CHIBA=067047  CHIOB=133404  CHKRN=156347  CHLAM=053402  CHLID=053320
CHLIM=051173  CHPAR=163312  CHPO =135164  CHREE=121164  CHRTD=120434
CHRUN=135212  CHSIZ=163345  CHVAR=126417  CINBT=147765  CLADB=003204
CLAMU=120541  CLCFI=034074  CLCLO=056652  CLENG=000112  CLERT=120723
CLEST=137164  CLFIE=034067  CLIDA=160050  CLLAM=053537  CLNRE=121050
CLODA=160214  CLP10=143756  CLPAG=120706  CLPUT=056530  CLR  =004004
CLRBM=053662  CLSEG=120661  CLWIN=120656  CLXMS=113015  CMBAB=054601
```

```
CMCHT=170520  CMCLG=116077  CMCPU=167577  CMLTS=156713  CMMCL=164746
CMODI=000104  CMOVB=067730  CMTRA=056466  CNVER=157742  COAF =134135
COCOD=171301  COLAM=053503  COLDS=117736  COMEA=171657  COMEB=171757
COMEC=172036  COMED=172122  COMEX=172213  COMRT=145762  CONTX=033764
CONVC=144240  COOPT=114420  COPTD=062106  COPYB=047547  COSDA=044630
COSEG=000046  COSPO=042242  COSTA=044645  COUTL=121213  CPCR =000100
CPF11=120155  CPF12=120156  CPF41=120151  CPF42=120152  CPFL1=120155
CPFL4=120151  CPFR1=120145  CPFR2=120146  CPFRS=120145  CPRIM=037777
CPRIS=000016  CPTSL=157150  CRBIN=125316  CREME=156030  CRET =000001
CRHEE=156106  CRHEO=156167  CRLAM=143452  CRTTC=000104  CSAR =000377
CSEGS=121065  CSGLE=000000  CSTSE=053756  CSWLG=120134  CTCW =000001
CTIBA=045740  CTMOD=046446  CTOBA=046057  CTR2M=141265  CTRDI=113632
CTRMA=141265  CTRME=161374  CUIDX=061046  CUMTA=125215  CWTTC=000104
CXRBG=110313  CXRBP=110264  CXRES=045666

D1000=164344  D1100=164347  D1200=164550  D1201=037242  D1225=044513
D1226=044516  D1227=044521  D1230=044524  D1232=044473  D1233=044476
D1234=044501  D1235=044504  D1300=164745  D1400=165146  D1500=165151
D1600=165152  D1700=165155  D2000=165304  D2100=165305  D2200=165466
D2300=165473  D2400=165514  D3CHA=060473  DALM1=035322  DAMON=055232
DATA =150257  DATAM=000001  DATIM=136052  DATSE=117540  DAYEA=055226
DAYES=055252  DBASE=122005  DBGET=110460  DBGPR=061047  DBGSE=117541
DBLDW=057204  DBPUT=110412  DBSPR=127445  DBSTW=057202  DBTRA=064002
DBUGS=117537  DCBX =000120  DCEMS=000204  DDEBU=037306  DDRTB=147045
DEBUG=061055  DECO =064147  DECOE=131462  DECOM=131403  DEDF1=114635
DEFEL=156171  DEFSG=124145  DELAM=143541  DEMFI=034030  DEMSV=133550
DENSI=171276  DEVBU=044570  DEVTL=171274  DF1  =034147  DF11 =034407
DF13 =034457  DF2  =034217  DF20 =034527  DF3  =034267  DF9  =034337
DFBLC=037602  DFDIL=035254  DFHDL=135072  DFPRM=123016  DFPTS=050364
DFRRT=034577  DH2  =000002  DH430=000070  DIA0 =066162  DIA1 =066211
DIA2 =066240  DICHA=145361  DIDLO=114611  DILBU=035334  DILEZ=000011
DILGS=000044  DILP1=036764  DILRT=132155  DIRTA=110060  DIRUN=133414
DISP1=000113  DISP2=000114  DISPE=034661  DISPR=132076  DLOF3=123161
DLOFU=053617  DLPTM=057133  DLSGF=124143  DLSTS=000000  DMLP1=036677
DMLPC=057113  DMMES=161661  DMS  =002004  DOCHA=145374  DPRIN=113540
DRISI=002000  DR2SI=002000  DRBI =000007  DRERR=147274  DRFEI=114322
DRTDE=117562  DRXAC=155750  DRXMS=113112  DSBAN=037600  DSIO =066267
DSI1 =066310  DSI2 =066331  DSSNM=000052  DSTAT=000067  DSTEN=120006
DT01R=036245  DT01W=036260  DT030=034721  DT037=034732  DT05R=036273
DT05W=036306  DT06R=036321  DT06W=036334  DT070=034743  DT075=034754
DT135=034776  DT140=034765  DT160=035007  DT285=035031  DT288=035020
DT300=035042  DT450=035053  DT460=035064  DTAPR=115533  DTBLC=037604
DTFU =131712  DTGPO=041412  DTHEA=044543  DTLI =000006  DTSLI=155643
DTTU =131710  DTXXX=035075  DTYYY=035106  DTZZZ=035117  DUMCA=127113
DUMF =134062  DUMM2=042204  DUMMY=041643  DURPR=122655  DURSU=122603
DUSE3=123230  DUSEL=053625  DUSSU=122536  DV000=163535  DV200=163652
DV300=163767  DV400=164170  DV500=164171  DV600=164336  DV700=164343
DVDIL=114764  DVMAS=136007  DVSTR=123011  DVTAB=055524  DWRIT=107327
DXCLU=055276

E1000=164347  E1100=164550  E1200=164745  E1300=165146  E1400=165151
E1500=165152  E1600=165155  E1700=165304  E2000=165305  E2100=165466
E2200=165473  E2300=165514  E2400=165515  E5CPU=066647  EBCHT=041634
EBPRT=133510  EBUFF=161647  ECAPD=106612  ECBR7=000001  ECH0 =106070
ECH1 =106100  ECH2 =106110  ECH3 =106120  ECH4 =106130  ECH5 =106140
ECH6 =106150  ECHSU=106533  ECLEA=000110  EDINB=050217  EDISP=000117
EDRSP=161672  EDSIB=066352  EDTRM=060605  EGXMS=153000  EILFN=000105
EILFZ=000115  EILLI=000106  EILSI=000113  EINP =000111  EIOBU=030356
EIOXT=044370  ELO3F=123157  ELOCK=000120  ELOF3=123116  ELOFF=053611
ELOFU=053614  ELON =053606  ELON3=123042  EMONI=137042  EMTY =004000
```

```
EN000=163652  EN200=163767  EN300=164170  EN400=164171  EN500=164336
EN600=164343  EN700=164344  ENCLE=000100  ENDBF=110233  ENDBU=044615
ENDDD=110136  ENDDT=110165  ENDHD=110107  ENDNT=110055  ENDOP=030330
ENDSP=044643  ENEN =044615  ENETS=150003  ENINI=000101  ENSIZ=001520
ENSYS=162122  ENT13=116025  ENTRC=051332  ENTSG=065055  EPAR =000112
EPRVT=133550  ERB  =001000  ERESP=161667  ERG  =005414  ERIOT=052040
ERRET=171543  ERREX=171557  ERRF =132150  ERRMO=047266  ERRNO=000000
ERRSP=161657  ERTDT=037043  ERUCL=055303  ESC50=073456  ESC50=073575
ESCDB=061063  ETMRT=165556  ETOMU=000107  ETOSM=000114  ETOU1=000103
ETOU2=000104  EUND =000102  EUSE3=123226  EUSEL=053622  EX112=055326
EX113=055322  EX21 =000116  EXEL =133642  EXPAN=140404

F0   =044543  F1101=037172  F1102=037176  F1136=037202  F1137=037206
F1142=037212  F1143=037216  F1144=037222  F1150=037226  F1151=037232
F1177=037236  F1200=037352  F1202=037246  F1203=037562  F1204=037252
F1205=037256  F1222=037262  F1223=037266  F1352=037356  F1600=044570
F1731=037272  F1UOI=036044  F1UOO=036107  F2166=037276  F2167=037302
F5DSG=000047  FAERO=163232  FAERR=163234  FALSE=147145  FATER=153217
FBFBU=147171  FBLCK=037602  FCPUL=167545  FCPUN=110233  FDABS=004215
FDFSE=044537  FDIBU=143627  FDID1=036022  FDIFE=143643  FDIFI=143663
FDPLX=000000  FDSTR=000072  FDVBU=146733  FENTR=150275  FEOL =000000
FFITA=120573  FFMTA=120565  FFSTA=120601  FFTAB=110055  FILUS=052175
FINDI=120245  FINET=171613  FINFD=143276  FIXC =065267  FIXC5=065466
FIXCR=120161  FIXOK=135022  FLBPA=000100  FLEAV=150340  FLV1S=143617
FLYTT=050052  FMAX =000034  FMEMC=172003  FNO  =135445  FPFCO=116705
FPL3A=036153  FPS41=031737  FPT2E=122003  FPTSL=050364  FR5TM=133236
FROME=073043  FRSEG=037042  FRSSZ=000026  FRUSE=136051  FS20 =037152
FS21 =037156  FS27 =037162  FS4  =037132  FS5  =037136  FS6  =037142
FS61 =037166  FS7  =037146  FSEMA=067470  FSERM=002003  FSF  =002413
FSSTA=140000  FUNCO=171277  FUNCT=145551  FUNIL=171642  FXCTA=065205
FXTAD=047365

G5BUF=002256  GAAC =053303  GAPP =053312  GBRKD=051116  GBSEG=002355
GBTIN=054700  GDBPA=054444  GDEBU=147507  GDEV3=122265  GDRXM=054302
GECHO=107157  GEPNE=145456  GERDV=055360  GET0 =063635  GET1 =063631
GET2 =063625  GET3 =063621  GET4 =063615  GET5 =063611  GET6 =063605
GETBI=047472  GETDA=051476  GETDW=115575  GETLA=122044  GETLS=072103
GETMB=156421  GETME=156254  GETPO=157767  GETS2=063600  GETSO=111617
GETVA=156304  GFIAC=133672  GFILI=133075  GLAPP=053311  GLCAC=053302
GLDN =044540  GMAIN=124023  GMUSI=130506  GNSTA=034000  GOMAG=142226
GOODR=126424  GPBUS=152124  GPERR=152461  GPFIN=152135  GPIBD=153301
GPIBH=153177  GPIBT=151507  GPIIM=145477  GPUPI=147303  GRTDA=060101
GRTNA=060105  GSEG =032003  GSGNO=060075  GSYS =146574  GTDF =135710
GTRT =047326  GTSLP=051067  GTSUB=153112  GUSE =146667  GUSEN=131070
GVAL =145405

HALF =000001  HASTA=000076  HDAY2=055255  HDAYE=055251  HDERC=000066
HDIF1=040145  HDIF2=040514  HDLC =145020  HDOF1=040003  HDOF2=040352
HDREC=147641  HDSIN=150414  HDSTA=147144  HDTM2=060413  HDTM3=060416
HFLGH=146751  HH2  =000000  HH430=000000  HHEAD=000006  HHIST=146744
HHMAX=000005  HIDTE=140136  HIGHT=161645  HIINT=150073  HILI1=040275
HILI2=040644  HISTI=146753  HISTO=146535  HISTS=137111  HOINT=147525
HOLI1=040124  HOLI2=040473  HOURS=055253  HRSTA=133620  HTEST=146743
HX21D=000015  HX21M=060000  HX21S=000016  HXDOK=177767  HXST =150301
HXTMO=177770

IACT5=133247  IB01I=037453  IB010=037470  IB4IN=112043  IB8IN=111620
IBACK=122200  IBMOV=074422  IBR3S=123277  IBRSI=053645  ICHAI=145276
ID01I=037372  ID010=037422  ID02I=037504  ID020=037534  ID10T=166316
ID12T=166316  IDBUS=000040  IDISP=000006  IDMO1=036555  IEDCH=045575
```

```
IEIDT=121255  IERRF=034117  IFM50=133206  IFSIZ=000004  IIDTA=121251
IIIOX=125604  IISTA=106162  IITAB=120504  ILL  =177777  ILUTA=126737
IMHDL=145162  IMODU=000003  IMREE=064560  IMTAB=120456  IMTRI=057237
IMTRO=057353  INBX2=060540  INIBD=161675  INISN=161756  INIT =000004
INMES=140102  INORO=121055  INSRP=121332  INTCH=102164  INTDF=033076
INTST=000103  IOACT=145614  IOAPD=110156  IOBUT=030331  IOREM=055153
IOTR =061770  IOUTL=175040  IOXTA=044360  IPRIV=060121  IRTCP=116076
IRTRY=000005  ISPHE=127635  ISTAB=120533  ISTCK=150364  ISYSE=132537
ISZRS=161656  IT13E=166145  ITB10=165557  ITB11=165677  ITB12=165720
ITB13=166145  ITE10=166071  ITE11=166104  ITE12=166123  ITE13=166140
ITIMT=120624  ITO50=133065  ITSLI=160155

KEXMC=122006  KGPIB=145261  KLRC1=172117  KREAD=134764

L5DSG=000051  LAINF=144156  LAMAR=125024  LAMDI=053576  LAMFU=142334
LAPLI=142340  LAPRO=143612  LAREA=144010  LBSEG=001000  LBYTC=000001
LDA1X=052123  LDA2X=052127  LDA3X=052133  LDAB =045401  LDAX =045351
LDEND=172205  LDRAD=171733  LDT6X=052077  LDTB =045373  LDTX =045356
LDVFC=150240  LDXOX=052147  LDXB =045365  LEGSC=120411  LEGSE=064171
LEOL =000004  LEV1 =124622  LEV14=124772  LGARR=145764  LHAST=000002
LIADD=127575  LIDTE=140037  LIHDL=135076  LIINT=000100  LIMCH=120444
LINAR=037027  LINKO=122661  LISCO=125714  LISTF=000101  LISTL=000117
LISTP=000077  LISTR=152602  LKEY =000000  LKICK=135313  LLAMC=144461
LMASK=060377  LMEM =000002  LMEM1=000002  LMEM2=000003  LNK1S=064737
LOADB=156672  LOADR=000200  LOBYT=052067  LOFFA=051533  LOG1 =146031
LOGDI=162027  LONAL=051540  LOOKA=150736  LOTH1=146533  LOTH2=146534
LOWAC=074236  LP1BU=166316  LPL3A=036235  LPOIN=135312  LPPUT=113503
LPTSL=050365  LRU  =121665  LTADB=045477  LTADX=045473  LTOUT=147245
LTSPR=157404  LWIAR=142355  LWR  =007002

M1TMR=165516  M5CPU=150472  M5SGS=150032  M88IW=144316  M8IBT=111466
M8IDR=144302  M8IIN=144622  M8INB=057723  M8INT=111353  M8OIN=111315
M8OTE=111232  M8OUT=057600  MABST=004131  MACD =153777  MAGTR=142277
MAINT=000075  MAKET=175160  MAMOD=000002  MAPSI=066352  MATA1=000002
MAX10=000120  MAX11=000021  MAX12=000151  MAX13=000151  MAXCA=000001
MAXEM=000004  MAXR =000115  MAXUS=000001  MBABP=054613  MBFDI=056106
MBPRV=050352  MBSPR=050352  MBSRE=114124  MBTER=151530  MBUSY=142312
MC144=055516  MCALL=064052  MCCLC=110234  MCCLD=136253  MCCO =137065
MCHAN=134252  MCLEA=000003  MCLGB=116100  MCLGF=116102  MCLGO=116104
MCLGP=116103  MCSTA=114000  MDEAB=134707  MDRIV=114307  MDRTB=147133
MDSEG=051105  MELRT=051702  MERRC=057446  MESBU=136454  MESMA=135446
MESSR=137352  MEXIT=064064  MFEIL=142361  MFIN =142100  MFINI=142335
MFRES=142606  MFXMS=113041  MGOTA=116105  MGTMR=055401  MIB10=002114
MIB11=002114  MIB12=002114  MIB13=002114  MIB14=002114  MIB15=002114
MIB16=002114  MIBU1=002114  MIBU2=002114  MIBU3=002114  MIBU4=002114
MIBU5=002114  MIBU6=002114  MIBU7=002114  MIBU8=002114  MIBU9=002114
MINBT=150442  MLAMU=052243  MLBRO=140200  MLCTA=137051  MLDBR=140634
MLDIL=065042  MLDLB=141350  MLDLM=141346  MLDSE=140455  MLGTM=054050
MLHEL=137771  MLIDF=054071  MLINI=140042  MLLBR=141122  MLLME=141120
MLOGI=054045  MLRUN=140125  MLSEN=140176  MLSTO=140122  MLVLO=044647
MLVSA=044645  MLVUL=044662  MMBRE=126647  MMEXY=051265  MMLE =052153
MOB10=000344  MOB11=000344  MOB12=000344  MOB13=000344  MOB14=000344
MOB15=000344  MOB16=000344  MOBU1=000344  MOBU2=000344  MOBU3=000344
MOBU4=000344  MOBU5=000344  MOBU6=000344  MOBU7=000344  MOBU8=000344
MOBU9=000344  MOCTB=063466  MODI1=140112  MODIN=140111  MODUT=140760
MOFIX=064257  MOLDN=044541  MOLDS=000516  MONIC=137051  MOSTI=054740
MOSTO=055037  MOTMI=055012  MOTMO=055125  MOTRI=055021  MOTRO=055115
MOURE=057700  MOUTB=150456  MOVIT=160024  MPAGE=060714  MPASE=060712
MPM4D=037046  MPRVT=150732  MRCPL=051703  MRLCL=065026  MRSEG=061175
MRTRG=051700  MRXRE=051701  MSGLE=146504  MSGMX=000474  MSIBB=066355
```

```
MSINI=137716  MTCLD=055432  MTDI1=035537  MTDO1=035601  MTFIE=035434
MUDMA=075211  MUNFI=064340  MXLAM=000040  MXLIN=037027  MXLPR=000002
MXMSG=163517  MXRET=066127  MXSIB=066153  MXX10=000000  MXX12=000000

N500C=132216  N500D=066417  N500M=066700  N500S=074035  N500T=074221
N5ADR=100000  N5DSG=000053  NACCO=061115  NALME=000005  NAMSE=037336
NAMTA=110003  NBLCK=037607  ND852=134042  NDEMF=034000  NFERR=153201
NFLAD=142306  NINIT=125145  NLP  =003000  NMAXU=061036  NN310=144514
NN5MS=000457  NNBRT=000004  NOBGP=000005  NONTT=050347  NOP  =000004
NRFSG=044643  NSLAR=142344  NSWPA=125074  NTRMS=000003  NU5PR=000003
NULDN=000010  NW5ST=136707  NW9ER=117460  NWORD=114366  NWREM=161655
NWSTA=161171  NXRTP=000034

O3CHA=060445  OCHAI=145305  OCT13=063541  OCTOB=154312  OCTOP=041622
ODUMP=121630  OERRF=034103  OFFES=137243  OFLDN=000507  OISIZ=046726
OISTA=106163  OKMON=137046  OLDH =134775  OLDL =134774  OLDN =000506
OLDSE=117542  OMSG =000102  ONESC=137263  ONTTM=050351  OOAPD=110243
OP2EN=173774  OPEND=172205  ORBIT=135451  ORERR=000001  OUTLI=116735
OVERL=123247  OVERR=100000  OXONC=111120

P2PAG=063711  P2XMS=113075  P3OST=112267  PAGE2=120324  PAGEF=120327
PAGPN=065217  PANEL=055351  PAR1 =142335  PAR2 =142336  PAR3 =142337
PARA1=142341  PARA2=142342  PARA3=142343  PCCS =123436  PCHSY=067560
PCOLD=063717  PCON1=113516  PCREG=000110  PDRIV=075320  PERIO=055250
PFEIL=114312  PFIXC=114004  PFLAM=143203  PFXMS=113032  PGAEN=146507
PGARR=146331  PICAC=134777  PIDRE=123457  PIERE=123456  PIKIC=135612
PILOA=135240  PIO01=040755  PIOCE=135232  PIOCM=075313  PIOFL=047431
PIOFS=047435  PIONP=073710  PIORE=075336  PIREL=135570  PIRES=135540
PIRET=133611  PISEF=135103  PISIZ=047144  PISTA=135316  PISTO=135273
PITIM=075323  PIUNL=135715  PIWKI=135661  PLDWO=057212  PLOTT=055163
PLXCL=055300  PM116=133775  PM160=134032  PMTRA=063702  PNW5S=074721
POFMO=116763  POFNM=117030  POFTO=147606  PONIO=112572  PPAGE=063707
PPWFA=063713  PPX3C=055273  PPXCL=055272  PREAD=134751  PRESY=063715
PRLOG=054722  PRSDC=147565  PRTCG=145436  PRTDR=061337  PRVTT=133510
PSTAT=000005  PSTWO=057206  PSTWR=175155  PT3CO=047534  PT3FU=125420
PT3MB=132354  PT3MC=123515  PT3UD=131232  PT5RS=074741  PTABL=113463
PTBAS=055221  PTERM=114136  PTLAM=142532  PTSIN=110003  PUTDW=115634
PUTPO=157747  PWFAI=123475  PWRFR=055302  PWRIT=134761  PX8CL=055260
PX9CL=055257  PXCLX=055272  PZCRA=145240

QDLAY=030361

R15ER=064205  R1IMA=121102  R1MEM=121110  R1SAV=121126  R3BUF=002736
R5BUF=002746  RACTB=147266  RANDO=122203  RBACK=122202  RBGET=110344
RBPUT=110364  RBUFF=141763  RBYTC=147271  RCALA=150114  RDATR=161646
RDB  =005001  RDF  =001011  RDMA =000014  READC=055305  READM=141576
RECSE=150611  REDOM=045555  REEIN=054002  REENT=064556  REESM=037122
REEUT=054004  REFBP=051106  REGBL=117544  REIDT=121257  REJEC=163400
RELNA=072417  RELOA=171626  RELRE=070636  REMCM=134162  RERRP=060770
RESAJ=050743  RESCF=161654  RESNA=072340  RESTA=123554  RESYS=117741
RETSG=114346  RETUA=114345  REW  =003404  RFDSG=150142  RFIMS=161664
RFIOX=125606  RFSGN=044642  RFSTB=044644  RFTS =147431  RHIFE=130042
RIDTA=121253  RIMAG=120142  RIOWA=074255  RKTAS=155623  RL5PD=072742
RLEGS=064214  RLOCA=161651  RLUTA=126741  RMAGT=125016  RMEMO=120162
ROBJE=131533  RPAR =037610  RPASS=037030  RPIT3=113605  RRDR =000000
RRS  =000002  RRT1 =034643  RRTS =000010  RRTWT=133434  RSAVE=120202
RSCLD=173730  RSGMO=071735  RSPHE=127637  RSPTA=124537  RSRES=045546
RSTAR=133626  RTBES=042507  RTCGP=052225  RTDIL=041737  RTDIR=061304
RTDYN=000065  RTERR=041701  RTIMT=120630  RTL2W=051726  RTLFI=037126
```

```
RTLMC=051561  RTLRE=051645  RTLRF=051734  RTLWF=051731  RTOFF=047342
RTON =047335  RTPNB=000063  RTRFA=042165  RTSLI=041775  RTSR =000006
RTTAB=055524  RTTS =000012  RUN  =007404  RWGET=110515  RWPUT=110534
RWRT1=042071  RWRT2=042110  RWRT3=042127  RWRT9=042146  RWSEG=122373


S000 =044371  S1101=001101  S1102=001102  S1105=001105  S1106=001106
S1107=001107  S1110=001110  S1112=001112  S1114=001114  S1117=001117
S1120=001120  S1121=001121  S1122=001122  S1123=001123  S1124=001124
S1127=001127  S1130=001130  S1131=001131  S1132=001132  S1133=001133
S1134=001134  S1135=001135  S1142=001142  S1147=001147  S1150=001150
S1151=001151  S1152=001152  S1153=001153  S1154=001154  S1155=001155
S1160=001160  S1161=001161  S1162=001162  S1163=001163  S1164=001164
S1165=001165  S1166=001166  S1171=001171  S1172=001172  S1177=001177
S1225=001225  S1226=001226  S1227=001227  S1230=001230  S1232=001232
S1233=001233  S1234=001234  S1235=001235  S1300=001300  S1301=001301
S1333=001333  S1334=001334  S1335=001335  S1336=001336  S1337=001337
S1340=001340  S1341=001341  S1342=001342  S1732=001732  S1733=001733
S1734=001734  S1735=001735  S1737=001737  S1740=001740  S1741=001741
S1742=001742  S1743=001743  S1744=001744  S1745=001745  S1746=001746
S1747=001747  S1750=001750  S1751=001751  S1752=001752  S25D =044401
S32D =044407  S33D =044435  S34D =044415  S3L13=060464  S3NOA=054240
S40D =044373  S41D =044376  S41FL=043511  S41FP=000146  S500E=044265
S500S=044151  S504 =000504  S505 =000505  S506 =000506  S507 =000507
S511 =000511  S512 =000512  S513 =000513  S514 =000514  S516 =000516
S521 =000521  S522 =000522  S523 =000523  S524 =000524  S525 =000525
S554 =000554  S555 =000555  S556 =000556  S557 =000557  S561 =000561
S563D=044427  S564D=044432  S570 =000570  S571 =000571  S572 =000572
S573 =000573  S5CPU=066647  S5ESC=050622  S5TSL=073737  SACLE=053330
SADTS=147260  SAGPA=112110  SAREG=000111  SAVE =156332  SBANK=037600
SBFIE=172204  SBMAX=000100  SBPRT=133436  SBUF =037601  SBYTC=147254
SCALL=153026  SCE06=130731  SCHPR=130061  SCOMT=114532  SCPOU=120702
SCPRO=132620  SCRET=147263  SCSFS=132770  SDATA=150247  SDUMM=044372
SEC2 =055256  SECON=055254  SEGAD=120164  SEGCH=121004  SEGIN=121416
SEGMO=071704  SEGSO=122120  SEGTX=043644  SELAR=120237  SEM60=037076
SEM61=037102  SEM62=037106  SEM63=037112  SEM64=037116  SEMDL=037346
SEMI1=037052  SEMI2=037056  SEMI3=037062  SEMI4=037066  SEMI5=037072
SEMPI=040710  SEMSE=001204  SENCT=134403  SET5N=052047  SETBF=050116
SETCL=055333  SETCP=136674  SETPT=123631  SETRT=120751  SETSI=163253
SET5T=044752  SFFUN=142736  SFTIM=142634  SG02S=000000  SG03S=110000
SG04S=110000  SG05S=022000  SG06S=110000  SG07S=110000  SG10 =043310
SG10S=000000  SG11 =043315  SG11S=000000  SG12 =043322  SG12S=140000
SG13 =043327  SG13S=056000  SG14 =043334  SG14S=110000  SG15 =043341
SG15S=110000  SG16 =043346  SG16S=150000  SG17 =043353  SG17S=000000
SG2  =043252  SG20 =043360  SG20S=000000  SG21 =043365  SG21S=000000
SG22 =043372  SG22S=110000  SG23 =043377  SG23S=110000  SG24 =043404
SG24S=140000  SG25 =043411  SG25S=140000  SG26 =043416  SG26S=110000
SG27 =043423  SG27S=000000  SG3  =043257  SG30 =043430  SG30S=122000
SG31 =043435  SG31S=122000  SG32 =043442  SG32S=110000  SG33 =043447
SG34 =043454  SG35 =043461  SG36 =043466  SG36S=110000  SG37 =043473
SG37S=000000  SG4  =043264  SG40 =043500  SG40S=164000  SG41 =043505
SG42 =043512  SG42S=110000  SG43 =043517  SG43S=110000  SG5  =043271
SG6  =043276  SG7  =043303  SGAND=066015  SGBRK=110001  SGFIX=115110
SGLEN=000000  SGMAD=001266  SGNUM=000054  SGOR =066017  SGPBS=145304
SGPIB=144614  SGWPE=134670  SGWPR=134664  SH2  =000000  SH430=000000
SHIST=147061  SHRSO=123365  SIBAP=066157  SIBBD=066154  SILFO=100000
SINSE=117036  SINTR=172243  SLAMC=144560  SLAMU=120537  SLCMO=154502
SLD1 =000005  SLDN =044533  SLRMO=047353  SLVTB=146507  SM1AB=110544
SM2DE=113401  SM2LE=110321  SM2TC=113704  SM3LE=110273  SM3OC=113514
SMABL=110451  SMAGP=112122  SMAGT=171307  SMBAC=113167  SMCAM=114517
```

```
SMCCL=113216  SMCHT=110373  SMCIL=114434  SMCLE=150327  SMCRE=113120
SMCRL=113675  SMCWR=113133  SMDDE=113317  SMDEC=113540  SMDOC=113442
SMDTD=113534  SMDYN=114012  SMEDI=111020  SMEND=110205  SMENT=110250
SMFAM=114474  SMFFL=153330  SMFIB=114051  SMFIL=117235  SMGCO=110656
SMGFI=123327  SMGPA=112130  SMKGP=113061  SMLEA=110332  SMMAI=137165
SMOCT=113473  SMOOP=145522  SMORE=113125  SMOUT=113230  SMOWR=113140
SMPER=113600  SMSCA=113174  SMSGP=112115  SMSPA=113663  SMSRC=111657
SMSTR=110232  SMTAC=114037  SMTBR=055466  SMTCI=113723  SMTCO=113715
SMTIM=112007  SMTMT=111720  SMTRA=056344  SMTRS=153672  SMTXF=114506
SMWIN=113735  SMXLE=110306  SMYES=110433  SNDBU=160310  SNDCP=161302
SNDRE=160455  SNDRF=160331  SNETS=150000  SNS  =006004  SNWP1=120157
SNWP2=120160  SNWPS=120157  SPARM=125357  SPBUF=147417  SPCOR=125252
SPITM=137017  SPLRE=064645  SPOOD=147237  SPOOS=147265  SPORT=132677
SPPR1=044616  SPRIM=140000  SPRIS=177762  SPRT1=042223  SPSG1=000045
SPSGL=000004  SPSTA=150672  SPTOP=054242  SPTOW=050203  SPT3P=054244
SPT3W=113612  SPTAB=044617  SRCSH=111631  SRDAT=000073  SREEN=065606
SRELE=123113  SREMO=122777  SRESE=123023  SRETR=056512  SRMMO=154500
SRRES=000072  SRRT1=034647  SRTER=110052  SRTON=110177  SS41I=065572
SSBUF=142234  SSCGE=047442  SSCLD=173652  SSDNU=127331  SSLEA=117154
SSMON=117057  SSSEG=000053  SSSLE=117153  SSSWP=064661  SSTPC=000003
STA0X=052103  STA3X=052107  STA4X=052113  STA5X=052117  STA6X=052073
STAB =045210  STABA=154424  STACH=172432  STADB=045273  STADX=045267
STAGP=110062  STAKO=172345  STARC=155616  STARE=172237  START=032017
STAX =045172  STBAC=050565  STFLG=153562  STL12=060357  STL13=060375
STMLE=065523  STMRS=055455  STMSY=135444  STOBA=154426  STOBY=052063
STOGP=110074  STORB=156727  STORE=126122  STOTA=163042  STPCN=000070
STSEG=120672  STSIN=041662  STSLI=121662  STTB =045205  STTIN=106165
STTX =045175  STUSP=060316  STWOR=147747  STXB =045202  STZ0X=052137
STZ3X=052143  SVRBL=052220  SVREI=071420  SWDEN=126611  SWLGO=120136
SWPBL=063674  SWPCO=063667  SWPLO=166062  SWPMA=063671  SWPMT=063673
SWPPA=063675  SWPRE=064777  SX1D =044443  SX2D =044446  SX3D =044451
SX4D =044454  SX5D =044457  SX6D =044462  SX7D =044412  SX8D =044420
SYSEV=032714  SYSST=000011  SYSSZ=002400

T1BEX=045114  T1XEX=045033  T206R=123466  T207R=123074  T2BEX=045112
T2C06=053650  T2P01=057652  T2P02=057655  T2P03=060006  T2P04=060013
T2P05=060020  T2P06=053651  T2P07=053630  T2XEX=045031  T3REP=122103
T5RST=074356  T8INP=057731  TADAD=042336  TADFP=172000  TADLP=172001
TAPEF=147067  TBASE=055226  TBLCK=037604  TBREA=106713  TBUSP=124444
TDBGE=110456  TDBPU=110410  TDGET=107025  TDLPH=172002  TECHO=106644
TELEN=110564  TELIN=110561  TER60=133365  TERMC=177777  TERMP=042033
TERMS=151675  TFDIS=142753  TFPF1=120137  TFPF2=120140  TFPFS=120137
TFWBA=065755  THISS=042470  TIMRT=042470  TIOBU=171764  TLID =142433
TLPRI=143700  TMRTA=165515  TMRTE=165546  TNMCA=116101  TNPAG=142421
TOFEN=052154  TOOPC=154511  TOPOF=065536  TPF11=120153  TPF12=120154
TPF41=120147  TPF42=120150  TPFL1=120153  TPFL4=120147  TPFR1=120143
TPFR2=120144  TPFRS=120143  TPPAG=142405  TRAPD=131526  TRASE=150564
TRDLP=057110  TREMO=153072  TREPS=161660  TRFP =142441  TRNSE=122305
TRT1 =034213  TRT11=034453  TRT13=034523  TRT2 =034263  TRT20=034573
TRT3 =034333  TRT9 =034403  TRTLD=051611  TS3CO=071572  TSLAN=110051
TSLBR=050366  TSLCO=044141  TSLES=050376  TSLHA=050363  TSLHT=050362
TSLLO=050361  TSLLP=050406  TSLNE=050516  TSLNT=044145  TSLPR=050416
TSLST=044135  TSLSY=050360  TSLTI=050456  TSLTU=050357  TSTBA=050563
TSTCH=145355  TSTDE=044677  TSTEN=044701  TSTOW=044675  TSTSH=115364
TTGET=107027  TTMWA=050350  TUSON=126026  TXUND=000002  TYENT=106170


UADDR=134655  UASTR=134622  UBUSY=134660  UD11T=134714  UDF01=041351
UDFUN=134054  UDI01=041252  UDMA =075244  UDMO1=036613  UDO01=041270
UDR01=042355  UDTMO=075302  UELGN=153454  UESEM=037332  UFIN =134663
```

```
ULDN =000505  UNINO=171300  UPDIE=114552  UPDMB=156347  URERT=037566
URT01=131740  US0  =062360  US0X =062360  US1  =062507  US1X =062507
US2  =062643  US2X =062643  US3  =062735  US3X =062735  US4  =063017
US4X =063017  USIOX=125241  USVRE=071425  UWT11=075260

VDD01=001716  VDD02=001717  VDD03=001720  VDD04=001721  VDUSB=111607
VDUST=111575  VSXGE=047455

W1IMA=121114  W1MEM=121122  W1SAV=121161  WAIF =134065  WBACK=150407
WBC  =000005  WBFSZ=000004  WBUFF=141770  WCHL =000004  WDADR=000006
WDCR =000017  WDEA =000006  WDMA =000015  WDUM =000003  WFLDN=044536
WFPAK=000001  WFREE=000000  WHLIM=000003  WIMBA=121214  WLLIM=000002
WLPAK=000000  WMESS=141672  WNBUF=000000  WNFRE=000001  WNPAK=000001
WOBJE=131711  WORDP=156530  WPAR =037614  WPCR =000001  WPNUM=000002
WPOSS=000002  WPRI =000006  WPROC=000003  WPVER=177777  WRDIL=114715
WRMHE=155773  WRT  =003012  WRT11=042261  WRT13=042300  WRT20=042317
WRTC =000011  WSAR =000003  WSCA =000007  WSCOM=000004  WSEG =064440
WSEGX=064524  WSEQ =000005  WSLDN=044534  WTADR=000071  WTCR =000007
WTDR =000005  WTM  =001414  WTTC =000013  WTYP =000004  WULDN=044535
WXSAV=121201

X211 =130072  X2110=130667  X2112=130733  X2114=130220  X2116=130355
X2118=130177  X2119=130277  X212 =130401  X2120=130306  X2121=130326
X212S=126275  X213S=126306  X217 =130560  X218 =130770  X219 =131012
X21AA=126620  X21BR=127346  X21C =127360  X21CH=127303  X21CL=127054
X21CN=127130  X21DC=127256  X21ER=126542  X21F1=041046  X21GC=127633
X21IN=127442  X21LV=127433  X21PG=126414  X21RD=127254  X21RE=127003
X21RT=130743  X21SH=130251  X21TO=127713  X21UT=126554  X22SI=000020
X2A00=125715  X2A02=125735  X2A05=125755  X2A10=125775  X2A11=126015
X2A12=126035  X2AST=126055  X2ATL=126075  X2E01=131232  X2GET=130027
X2LL =126115  X2LL1=126116  X2LL2=126117  X2LL4=126120  X2LL5=126121
X2LMA=000017  X2NEW=126270  X2OLD=126271  X2RAC=126572  X2RBY=126574
X2S01=131032  X2SBY=126570  X2SND=126372  X2SSS=126437  X2STA=126502
X321I=060443  X321T=060441  X32ST=060437  X3T12=060515  X3T13=060517
X5   =044265  X5DCN=074750  X5MCS=075062  X9CL0=055262  X9CL1=055263
X9CL2=055264  X9CL3=055265  X9CL4=055266  X9CL5=055267  X9CL6=055270
X9CL7=055271  XALOF=000000  XBADM=154430  XBCKT=166516  XBLDN=000010
XBTLO=142054  XCCTA=125002  XCFUN=000000  XCHIO=174215  XCLNU=055274
XCLUN=055275  XCORM=172727  XCREA=050330  XCSEG=121077  XDASA=000010
XDHOM=113106  XDILD=035352  XDILF=035331  XEPT3=136672  XERR =153214
XFIN =142122  XFPGN=174751  XFUNC=037606  XGETA=174423  XGETB=067707
XGTDF=050141  XHBUF=000004  XHBYT=000003  XHDNX=000000  XHIST=147030
XIBNX=000004  XID10=000012  XID11=000012  XID12=000012  XID13=000004
XINIP=174706  XINIT=000107  XINST=133771  XIT05=133060  XLEV =000040
XLEVB=000050  XLEVL=000005  XLINK=117015  XLPT3=137777  XMBUS=142332
XMCHK=134100  XMEMC=174747  XMLIS=000003  XMPAT=147220  XMREM=134032
XMSGA=112772  XMSGY=113165  XNOBP=000004  XOFFE=073562  XOFTR=107203
XONCH=111027  XONES=073551  XONRE=111144  XONWR=111177  XOOAP=110252
XOUTL=175042  XP131=035327  XPFAD=112771  XPHYS=174763  XPION=067520
XPPRU=062343  XREEN=000106  XRELE=114106  XRERR=153207  XRESE=000002
XRETR=000105  XRSTA=044710  XRTDS=063242  XRTEN=063244  XRTSL=160157
XS5ES=050610  XSBPR=110111  XSDAT=147336  XSEGS=043240  XSERR=153213
XSETB=050073  XSETC=136675  XSGRT=043245  XSHDR=150316  XSPT3=114000
XSSND=147620  XSTAT=171275  XSTBA=050556  XSTDF=050160  XSWTP=171734
XTAPR=000020  XTARG=000013  XTASG=000030  XTBM0=000032  XTBM1=000033
XTBM2=000034  XTBM3=000035  XTBMA=000031  XTBMB=000034  XTBRG=000017
XTCHN=000000  XTCMS=000006  XTCNT=000025  XTDRG=000014  XTFRG=000012
XTHOM=000027  XTIME=000006  XTLRG=000015  XTLX =060162  XTMEM=000004
XTMMX=000005  XTMRG=000013  XTOPO=065555  XTPRG=000010  XTPRT=000003
XTRAN=063736  XTRDL=144001  XTRNS=122276  XTRSG=000031  XTRTA=000002
```

XTSAD=000023 XTSBF=000024 XTSBK=000023 XTSRG=000016 XTSTA=000001
XTSTC=126576 XTSTS=000007 XTTAP=000021 XTTCN=000026 XTTRG=000012
XTUBF=000022 XTXRG=000011 XWT05=063214 XXRPT=177777

YENDC=163535 YGETA=174421 YIBNY=000002 YOUHA=137735 YTRNS=065531

ZOPHY=063321 ZOUSR=063372 Z2PHY=063325 Z2USR=063376 ZBDIS=030356
ZBGET=113311 ZBINI=113252 ZBREL=113401 ZCRAS=145413 ZEROP=175134
ZSREL=123110 ZXCHK=147316 ZXCOM=113005 ZXRES=113050 ZXRRS=113057
ZXS12=063446 ZXS13=063454 ZXTRS=113066

%% :: Part Two Listing

N500.0000 L= SUPERLISTING VSX ..... = SIII-VSX VERSION J = **S3VS

GENERATED : 18.19.27 20 NOV 1984

```
176000  %%%%%%%%%%%  M A C R O E S  -  S I N 1  -  G E N  %%%%%%%%%%%%%%%%%%%%
176000
176000  %%      MACROES USED ON SIN1-GEN
176000  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
176000
176000
176000  % MACRO FOR "DF" DATAFIELDS:
176000
176000   )MCDEF DFFLD $NUM
DF$NUM , 0;0;*-2;0;0;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0;RWRT$NUM ;
        0;0;0;0;0;0;0;0
TRT$NUM , LDA *3; JMP I *1; COMMO; DF$NUM
]
176000
176000   "
176000
176000   % MACRO FOR TERMINAL INTERFACE INPUT DATAFIELDS
176000   % THE DRIVER USED IS T E L I N
176000   )MCDEF TELDF $NAM,$HNO
        0;-1;5;0;0;0;0
        TETTO;-3;-3
        $HNO ;TELIN;TELEN
$NAM . 0;0;*-2;110000;0;0;IORES
        TRGET;TEXIT;CLBUF;0;0
        BUF;40+40;0;0;40+40;0
%%"-8S3C
%%XBU=BUF+40;)KILL BUF;BUF=XBU;)KILL XBU
"8S3C
BUF=*; )KILL BUF; *+40/
"
]
176000
176000   )MCDEF TTUT $NAM,$HNO
        0;1;TTOMR;0;-10
        $HNO ;DWRIT;DWRIT
$NAM W, 0;0;*-2;110000;0;0;IORES
        TTPUT;DMOUT;CLBUF;$NAM R;0
        BUF;40+40;0;0;40+40;0
        40;0;0
"8STRN
        0;0;0;0
        0;0;0;0;0;0;0;0;0;0
"8S3C
BUF=*; )KILL BUF; *+40/
"
]
176000
176000
176000   % MACRO FOR TERMINAL-ACCESS-DEVICE (TAD)
176000   )MCDEF BADDV $NAM
        0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
        0;-1;0;0;0;0;33;0;3;410;0;ECHO;BRKO;0;0;0;0;0;0;0
BD$NAM R,  0;0;*-2;112020;0;0;IORES;BDGET;TEXIT;CTIBA;BD$NAM W;0
        0;0;0;0;0;0;0;0;TAD$NAM ;BFI$NAM ;0;0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;0
```

```
       0;0;BDTOU;0;-3;0;0;0
BD$NAM W, 0;0;*-2;112020;0;0;IORES;BDPUT;BDOUT;CTOBA;BD$NAM R;0
       0;0;0;0;0;0;0;0;0;0;0;0;BDROU;0;0;0
       0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
]
176000
176000
176000  )MCDEF HDDMA $LNR,$HNO,$ICTR,$OCTR

       $ICTR ;0;0;0
       0;0;0;0;0;0
       0;0;0;0;0;0;0;0
       0;0;0;0;0;0
       0;0;HDMID;HDIBU;HDIFI;HDIFE
       HITMR;0;-6;IOX $HNO ;CTRID;0
HDMI$LNR , 0;0;*-2;202;0;0;RETRA;0;0;HITRN;0;0

       $OCTR ;0;0;0
       0;0;0;0;0;0
       0;0;0;0;0;0;0;0
       0;0;0;0;0;0
       0;0;HDMOD;HDOBU;HDOFI;HDOFE
       HOTMR;0;-12;IOX $HNO ;CTROD;0
HDMO$LNR , 0;0;*-2;202;0;0;RETRA;0;0;HOTRN;0;0

]
176000
176000
176000  % MACRO FOR BSC/DMA - IO-DATAFIELD
176000
176000  )MCDEF HDIO $LNR

       0
HDFI$LNR , 0;0;*-2;12200;0;0;IORES
       TEXIT;TEXIT;TEXIT;HDFO$LNR ;0
       0;0;0;0;0;0
       1303+XHDNX;1305+XHDNX;0

       0
HDFO$LNR , 0;0;*-2;12200;0;0;IORES
       TEXIT;TEXTIT;TEXIT;HDFI$LNR ;0
       0;0;0;0;0;0
       1304+XHDNX;1306+XHDNX;0

YHDNX=XHDNX+4;)KILL XHDNX;XHDNX=YHDNX;)KILL YHDNX

]
176000
176000
176000
176000  % MACRO FOR SIN/SIN CHANNEL INPUT DATAFIELD
176000
176000  )MCDEF CHIN $NR,$LNR,$DNR
"8ECBR
       0;0;0;0;0;0;0;0        % ECHO AND BREAK TABLE 7
       0;0;0;0;0;0;0;0

       0;0;0;0;0;0;33
       0;0;0;4;ECHO;0;0;CMO$LNR ;0;0;0;0;$NR -1
S$NR $LNR R, 0;0;*-2;116000;0;0;CIORE
```

```
        SSTI;TEXIT;CLSBI;S$NR $LNR W;0
        0;*;0;0;0;0
        0;0;BC$NR $LNR ;BF$NR $LNR ;0;0
        0;0;0;0;0;0;0;0
        0;0;0;RU$NR $LNR ;0;0;0;0;0
        0;0;0;0;0;0
"-B$DNR L$LNR  -N$DNR L$LNR ;
DR$NR $LNR , 0;0;*-2;0;0;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0;CR$NR $LNR ;
        0;0;0;0;0;0;0;0
TR$NR $LNR , LDA *3; JMP I *1; COMMO; DR$NR $LNR ;
"

]
176000
176000   "CXCPU
176000   % MACRO FOR GENERAL TERMINAL 1-99
176000
176000   )MCDEF GTERM $NAM,$HNO
        BAK$NAM ;$HNO ;0;1
DT$NAM R, 0;0;33;110040;-1;44005;IORES

        0;$HNO +4;0;0
DT$NAM W, 0;0;*-2;110040;0;0;IORES
XYZ=NTRMS; )KILL NTRMS; NTRMS=XYZ+1; )KILL XYZ
]
176000
176000   % MACRO FOR GENERAL TERMINAL  100-128
176000
176000   )MCDEF GXTRM $NAM,$HNO
        BK$NAM ;$HNO ;0;1
T$NAM R, 0;0;33;110040;-1;44005;IORES

        0;$HNO +4;0;0
T$NAM W, 0;0;*-2;110040;0;0;IORES
XYZ=NTRMS; )KILL NTRMS; NTRMS=XYZ+1; )KILL XYZ
]
176000   "
176000
176000   % MACRO FOR SIN/SIN CHANNEL OUTPUT DATA FIELD
176000
176000   )MCDEF CHUT $NR,$LNR,$DNR
        0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;0;ECHO;0;0;CMO$LNR ;BUSIZ+BUSIZ;0;0;0;$NR -1
S$NR $LNR W, 0;0;*-2;116000;0;0;CIORE
        SSTO;TEXIT;CLSBO;S$NR $LNR R;0
        0;0;0;0;0;0;RU$NR $LNR ;0;0;0;0;0;0;0
        0;0;0;0;0
"-B$DNR L$LNR  -N$DNR L$LNR ;
DW$NR $LNR , 0;0;*-2;0;0;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0;CW$NR $LNR ;
        0;0;0;0;0;0;0;0
TW$NR $LNR , LDA *3; JMP I *1; COMMO; DW$NR $LNR ;
"

]
176000
```

```
176000  % MACRO FOR SIN/SIN LINE DATA FIELDS
176000
176000  DH2=2; DH430=70; SH2=0; SH430=0; HH2=0; HH430=0; CH2=0; CH430=0
176000  )MCDEF CTLIN $LNR,$XINR,$XONR,$TBS,$SND,$REC,$TIM,$CTR,$VNT,$CS,$OCTR,$IIN,$LB,$LP,$SP
        $CTR ;$TIM ;0;-3
        IOX $XINR ;$CS TIDR;$CS TIDR
TTIL$LNR , 0;0;*-2;2;0;0;IORES
        0;$CS RWAI;0;TTUL$LNR ;$IIN ;$LB ;0;0;0;0
        0;0;0;4
        $CS H2;$CS H430;0;$LP ;

        $OCTR ;
        IOX $XONR ;$CS TSDR;$CS TODR
TTUL$LNR , 0;0;*-2;2;0;0;IORES
        0;$CS SWAI;0;TTIL$LNR ;$CS RSVI;0;0;CMO$LNR ;0;1
        $SP ;


        14;-4;0;0;0;0;0;0
        -1;0;0;0;0;0;TISI
        0;-3;0;0;0
CMO$LNR ,  0;0;0;0;0
        0;0;0;0;0;TTIL$LNR ;$CS SINI;$CS BSIN
        *;0
        0;0;0;0
        $SND ;$TBS ;$REC ;$VNT ;
        *;0;0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;0;0;SCOM$LNR ;RCOM$LNR ;
        0;0;0;0
        0;0;$LNR ;0
        0;0;0;0
        0;0;0;0
        0;0;COTA$LNR ;2;0;AKMCS
IDA$LNR ,  MXCC
"C01L$LNR ;S01$LNR R
"C02L$LNR ;S02$LNR R
"C03L$LNR ;S03$LNR R
"C04L$LNR ;S04$LNR R
"C05L$LNR ;S05$LNR R
"C06L$LNR ;S06$LNR R
"C07L$LNR ;S07$LNR R
"C08L$LNR ;S10$LNR R
"C09L$LNR ;S11$LNR R
"C10L$LNR ;S12$LNR R
"C11L$LNR ;S13$LNR R
"C12L$LNR ;S14$LNR R
"C13L$LNR ;S15$LNR R
"C14L$LNR ;S16$LNR R
"C15L$LNR ;S17$LNR R
"C16L$LNR ;S20$LNR R
"
MXCC=*-IDA$LNR -2
XXY=AKMCS;)KILL AKMCS;AKMCS=XXY+MXCC+1;)KILL XXY

COTA$LNR =*
"C01L$LNR ;CDVNR;0; "B01L$LNR ;*-2/100000+CDVNR;0
"C01L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C02L$LNR ;CDVNR;0; "B02L$LNR ;*-2/100000+CDVNR;0
"C02L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C03L$LNR ;CDVNR;0; "B03L$LNR ;*-2/100000+CDVNR;0
```

```
"C03L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C04L$LNR ;CDVNR;0; "B04L$LNR ;*-2/100000+CDVNR;0
"C04L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C05L$LNR ;CDVNR;0; "B05L$LNR ;*-2/100000+CDVNR;0
"C05L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C06L$LNR ;CDVNR;0; "B06L$LNR ;*-2/100000+CDVNR;0
"C06L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C07L$LNR ;CDVNR;0; "B07L$LNR ;*-2/100000+CDVNR;0
"C07L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C08L$LNR ;CDVNR;0; "B08L$LNR ;*-2/100000+CDVNR;0
"C08L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C09L$LNR ;CDVNR;0; "B09L$LNR ;*-2/100000+CDVNR;0
"C09L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C10L$LNR ;CDVNR;0; "B10L$LNR ;*-2/100000+CDVNR;0
"C10L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C11L$LNR ;CDVNR;0; "B11L$LNR ;*-2/100000+CDVNR;0
"C11L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C12L$LNR ;CDVNR;0; "B12L$LNR ;*-2/100000+CDVNR;0
"C12L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C13L$LNR ;CDVNR;0; "B13L$LNR ;*-2/100000+CDVNR;0
"C13L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C14L$LNR ;CDVNR;0; "B14L$LNR ;*-2/100000+CDVNR;0
"C14L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C15L$LNR ;CDVNR;0; "B15L$LNR ;*-2/100000+CDVNR;0
"C15L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
"C16L$LNR ;CDVNR;0; "B16L$LNR ;*-2/100000+CDVNR;0
"C16L$LNR ;XDVNR=CDVNR;)KILL CDVNR;CDVNR=XDVNR+1;)KILL XDVNR
";)KILL MXCC
COTA$LNR +100/
]
176000
176000
176000
176000
176000
```

```
176000  )LINE
176000  %=============================================================================
176000  %       S I N 1 - G E N
176000  %=============================================================================
176000
176000
176000  %=============================================================================
176000  % 28.1        G E N E R A T I N G   P A R A M E T E R S
176000  %
176000  % SINTRAN III SYSTEM GENERATING PARAMETERS
176000  %
176000
176000  %----PAGE DISPLACEMENT FROM BEGINNING OF SINTRAN:DATA------------
176000
176000  5FVFS=144                       % FILESYSTEM SAVE-AREA
176000  5FVRL=502                       % RT-LOADER
176000  5FVSP=534                       % SPOOLING
176000  5FCRL=32+77                     % ND-NET FILECOPY
176000  5FVER=136                       % ERROR PROGRAM
176000  5FVOP=77                        % COMMAND SEGMENT
176000
176000  %---------------------------------------------------------------
176000
176000  5NASE=517
176000
176000  FLBPA=100                       % FIRST LOGICAL PAGE IN BACKGROUND SEGMENT
176000
176000  "5BDIS+8BD1 -55CDK -5GDIS -5WDIS
176000  %  MAIN SWAPPING DEVICE:   BIG DISC ON CONTROLLER 1
176000  5DSK=1100
176000  5ABLP=2
176000  5BLST=400
176000  5BL01=0
176000  8ZBDI
176001  "
176001  SEMSE=1204
176001  99MGT
176002  XDASA=10
176002  MACD=153777
176002  N5ADR=100000
176002
176002  BUF=0
176002  99CAC
176003  CURRR=*
176003  "     % 30/60/90 MB DISC (DSKTYPE=40000)
176003  CURRR/; )KILL CURRR
176003  "
176003
176003  LIDTE=140040-1                  % IDENT CODE OF TERMINAL 65 (-1)
176003  HIDTE=140137-1                  % IDENT CODE OF TERMINAL 128 (-1)
176003
176003  % DEFAULT VALUES:
176003  SPSGL;"SPSGL;SPSGL=4;"          % LENGTH OF SPOOLING QUEUE SEGMENT
176004  LBSEG;"
176005  LOADR;
176006  IDBUS;"IDBUS;IDBUS=40;"
176007  TERMC;"TERMC;TERMC=-1;"
176010  MIBU1;"MIBU1;MIBU1=2114;"
176011  MOBU1;"MOBU1;MOBU1=344;"
176012  MIBU2;"MIBU2;MIBU2=2114;"
```

```
176013  MOBU2;"MOBU2;MOBU2=344;"
176014  MIBU3;"MIBU3;MIBU3=2114;"
176015  MOBU3;"MOBU3;MOBU3=344;"
176016  MIBU4;"MIBU4;MIBU4=2114;"
176017  MOBU4;"MOBU4;MOBU4=344;"
176020  MIBU5;"MIBU5;MIBU5=2114;"
176021  MOBU5;"MOBU5;MOBU5=344;"
176022  MIBU6;"MIBU6;MIBU6=2114;"
176023  MOBU6;"MOBU6;MOBU6=344;"
176024  MIBU7;"MIBU7;MIBU7=2114
176025  MOBU7;"MOBU7;MOBU7=344
176026  MIBU8;"MIBU8;MIBU8=2114
176027  MOBU8;"MOBU8;MOBU8=344
176030  MIBU9;"MIBU9;MIBU9=2114
176031  MOBU9;"MOBU9;MOBU9=344
176032  MIB10;"MIB10;MIB10=2114
176033  MOB10;"MOB10;MOB10=344
176034  MIB11;"MIB11;MIB11=2114
176035  MOB11;"MOB11;MOB11=344
176036  MIB12;"MIB12;MIB12=2114
176037  MOB12;"MOB12;MOB12=344
176040  MIB13;"MIB13;MIB13=2114
176041  MOB13;"MOB13;MOB13=344
176042  MIB14;"MIB14;MIB14=2114
176043  MOB14;"MOB14;MOB14=344
176044  MIB15;"MIB15;MIB15=2114
176045  MOB15;"MOB15;MOB15=344
176046  MIB16;"MIB16;MIB16=2114
176047  MOB16;"MOB16;MOB16=344
176050  DR1SI;"DR1SI;DR1SI=2000;"
176051  DR2SI;"DR2SI;DR2SI=2000;"
176052  5SSSZ;"5SSSZ;5SSSZ=5;"
176053  GNSTA;"GNSTA;GNSTA=5SSSZ@12+BGSYS;"
176054
176054  "8LAMU;MXLPR; "MXLPR;MXLPR=2        % NUMBER OF LAMUS PER PROGRAM
176055  '8LAMU;MXLAM; "MXLAM;MXLAM=40       % NUMBER OF LAMUS IN THE SYSTEM
176056
176056  "SLP1;SLD1; "           % SPOOLING INDEX 1 DEFAULT LOG NO 5
176057  "          % SPOOLING INDEX 2 DEFAULT LOG NO 15
176057
176057  CCNO
176060  "          % 6K RTCOMMON IF 12 SIBAS PROCESSES
176060  "          % 3K RTCOMMON IF 6 SIBAS PROCESSES
176060  "          % 2K RTCOMMON IF 3 SIBAS PROCESSES
176060  "                % ELSE 0 K RTCOMMON
176060
176060  NULDN;"NULDN;NULDN=10;"        % NUMBER OF USER RESERVED LOGICAL  DEVICE NUMBERS
176061                                 % LDN'S STARTS FROM 2300
176061
176061  XTIME;"XTIME;XTIME=6;"         % NUMBER OF EXTRA ENTRIES IN TIMER TABLE
176062  XID10;"XID10;XID10=12;"        % NUMBER OF EXTRA ENTRIES IN EXT. IDENT TABLE, LEVEL 10
176063  XID11;"XID11;XID11=12;"        % NUMBER OF EXTRA ENTRIES IN EXT. IDENT TABLE, LEVEL 11
176064  XID12;"XID12;XID12=12;"        % NUMBER OF EXTRA ENTRIES IN EXT. IDENT TABLE, LEVEL 12
176065  XID13;"XID13;XID13=4;"         % NUMBER OF EXTRA ENTRIES IN EXT. IDENT TABLE, LEVEL 13
176066
176066  8RTN;"              % 24 RT DESCRIPTIONS
176067  8SGN;"              % 40 SEGMENTS
176070  '8XON;8RON                     % READER-ON IF X-ON
176071  '8XMSG;99ENS;8ALME;"           % ENTSG AND FIXC5 (ALLOCATE MEMORY AREA) WHEN XMSG
176073  CADEV;"CADEV;CADEV=700;"
```

```
176074  8IOXT;"8IOXT;8IOXT=10;"
176075  NALME;"   % DEFAULT MAX # OF MEMORY BUFFER TO BE ALLOCATED
176076  "
176076  9POFS;
176077  XIBNX=0; YIBNY=0
176077  XHDNX=0
176077  "-8ALOA;XALOF=0
176077  "8XON;8RON
176100  "8N500;8EXAD 8STRN          % ALWAYS EXTENDED ADDRESS MODE AND INSTR/OUTST WHEN ND-500
176101  "
176101  %%%    LIBRARY MARKS TO EASY REFERENCES TO MAGTAPE-FIELDS
176101
176101  % CONVERSION FROM 'OLD' TO 'NEW' FOR CONTROLLER 1
176101  "99TM1+99HM1+99SM1;8MT1;"
176102  %CONVERSION FROM 'OLD' TO 'NEW' FOR CONTROLLER 2
176102
176102  % TO ENSURE 8MT3
176102
176102
176102  "8N500; 8CP51; *-1/
176102
176102  "BAD01
176102   BADAD; *-1/
176102
176102  "-8N502 BADAD
176102  BADM=110001
176102  "
176102  % ========================================================================
176102
176102  %  FIRST SYSTEM AND BACKGROUND SEGMENTS (FOR TERMINAL 1)
176102  BSGM=22045
176102
176102  % ========================================================================
176102  "8DILG
176102  XYS=BSGM;)KILL BSGM;BSGM=XYS+401;)KILL XYS
176102  "SLP1
176102  XYS=BSGM;)KILL BSGM;BSGM=XYS+401;)KILL XYS
176102  "8COSP
176102  XYS=BSGM;)KILL BSGM;BSGM=XYS+401;)KILL XYS
176102
176102  "8PR01
176102  XYS=BSGM;)KILL BSGM;BSGM=XYS+1002;)KILL XYS
176102  "8N500
176102  XYS=BSGM;)KILL BSGM;BSGM=XYS+401;)KILL XYS
176102  "8RFAC; XYS=BSGM; )KILL BSGM; BSGM=XYS+401; )KILL XYS;
176102  "
176102
176102  "8DB01+8DB02+8DB03+8DB04+8DB05+8DB06+8DB07+8DB08+8DB09
176102  XYS=BSGM; )KILL BSGM; BSGM=XYS+401; )KILL XYS
176102  "
176102
176102  5FCOM=BSGM
176102
176102
176102  * **********
176102
176102   -8S3C
176102   ORAD/110000
000030
000030
```

```
000030  GNSTA/
034000  *-110000; )ZERO
034000
034000  %=============================================================================
034000  % 28.2        S T A N D A R D   D A T A F I E L D S
034000  %
034000  % MONITOR WORKING FIELD FOR NON-DEMAND
034000
034000  NDEMF=*; *+30/
034030
034030  % DATAFIELDS
034030
034030  DEMFI, 0;0;*-2;0;0;0;0;0
034040         0;0;0;0;0;0;0;0
034050         0;0;0;0;0;0;0
034057         0;0;0;0;0;0;0;0
034067
034067  CLFIE, 0;0;*-2;2              % SWAPPING SEMAPHORE
034073
034073
034073         ENT13
034074  CLCFI, 0;0;0;0;0;0;ICLK       % CLOCK
034103
034103  % ERROR MESSAGE INTERNAL DEVICE
034103
034103  OERRF, 0;0;*-2;110002;0;0;IORES;PUTW;OSTDV;CLBUF;IERRF;0
034117
034117  IERRF, 0;0;*-2;110002;0;0;IORES;GETW;ISTDV;CLBUF;OERRF;0
034133         BUF;50;0;0;50;0;0;0;5;0;-1;0
034147  "-8S3C
034147  %%XBU=BUF+50; )KILL BUF; BUF=XBU; )KILL XBU
034147  "
034147
034147  %=============================================================================
034147  % 28.3        D F   D A T A F I E L D S
034147  %
034147  % RWFILE FIELD
034147
034147  DFFLD 1
034147  DF1, 0;0;*-2;0;0;0;0;0
034157         0;0;0;0;0;0;0;0
034167         0;0;0;0;0;0;0;0
034177         0;0;0;RWRT1;
034203         0;0;0;0;0;0;0;0
034213  TRT1, LDA *3; JMP I *1; COMMO; DF1
034217
034217  DFFLD 2
034217  DF2, 0;0;*-2;0;0;0;0;0
034227         0;0;0;0;0;0;0;0
034237         0;0;0;0;0;0;0;0
034247         0;0;0;RWRT2;
034253         0;0;0;0;0;0;0;0
034263  TRT2, LDA *3; JMP I *1; COMMO; DF2
034267
034267
034267  "8M100
034267  89NMT=0
034267  "8MT1 -8XTRA
034267  DFFLD 3
034267  DF3, 0;0;*-2;0;0;0;0;0
```

```
034277          0;0;0;0;0;0;0;0
034307          0;0;0;0;0;0;0;0
034317          0;0;0;RWRT3;
034323          0;0;0;0;0;0;0;0
034333  TRT3, LDA *3; JMP I *1; COMMO; DF3
034337
034337
034337  "8F1U0
034337  99NFD=0
034337  "8FDI1+8BFD1 -8XTRA
034337
034337  DFFLD 9
034337  DF9, 0;0;*-2;0;0;0;0;0
034347          0;0;0;0;0;0;0;0
034357          0;0;0;0;0;0;0;0
034367          0;0;0;RWRT9;
034373          0;0;0;0;0;0;0;0
034403  TRT9, LDA *3; JMP I *1; COMMO; DF9
034407
034407
034407  "8DLP1+8DVE1 -8XTRA
034407
034407  DFFLD 11
034407  DF11, 0;0;*-2;0;0;0;0;0
034417          0;0;0;0;0;0;0;0
034427          0;0;0;0;0;0;0;0
034437          0;0;0;RWRT11;
034443          0;0;0;0;0;0;0;0
034453  TRT11, LDA *3; JMP I *1; COMMO; DF11
034457
034457
034457  "IBL01 -8XTRA
034457  DFFLD 13
034457  DF13, 0;0;*-2;0;0;0;0;0
034467          0;0;0;0;0;0;0;0
034477          0;0;0;0;0;0;0;0
034507          0;0;0;RWRT13;
034513          0;0;0;0;0;0;0;0
034523  TRT13, LDA *3; JMP I *1; COMMO; DF13
034527
034527
034527  DFFLD 20
034527  DF20, 0;0;*-2;0;0;0;0;0
034537          0;0;0;0;0;0;0;0
034547          0;0;0;0;0;0;0;0
034557          0;0;0;RWRT20;
034563          0;0;0;0;0;0;0;0
034573  TRT20, LDA *3; JMP I *1; COMMO; DF20
034577
034577
034577  "
034577  "8RFAC
034577  %     DF DATAFIELD FOR REMOTE FILE ACCESS FROM RT-PROGRAMS
034577  DFRRT, 0;0;*-2;0;0;0;0;0
034607          0;0;0;0;0;0;0;0
034617          0;0;0;0;0;0;0;0
034627          0;0;0;RTRFA
034633          0;0;0;0;0;0;0;0
034643  RRT1,   LDA *+3; JMP I *+1; COMMO; DFRRT
034647  SRRT1, LDX *+11; IOF; LDA ,X 12; BSET ONE 160 DA; STA ,X 12; ION
```

```
034655          MON 135; JMP I *+1; RRT1; RTRFA
034661  "
034661  %
034661  %================================================================================
034661  % 28.4          M A S S - S T O R A G E  D A T A F I E L D S
034661  %
034661
034661
034661  %---------------------------------------------------------
034661  %
034661  %    DISC TYPE POINTER       ==> Optional
034661  %
034661  %            0       1       2       3       4       5       6       7
034661  DISPE,      0; DT010; DT033; DT066;      0;      0;      0;      0 % 00 - 07
034671          DT014; DT021; DT023; DT045;      0;      0;      0;      0 % 10 - 17
034701          DT030; DT037; DT070; DT075; DT140; DT135; DT160; DT288 % 20 - 27
034711          DT285; DT300; DT450; DT460; DTXXX; DTYYY; DTZZZ;      0 % 30 - 37
034721
034721  %---------------------------------------------------------
034721  %
034721  %  DISK LAYOUT:
034721  %       SECWO, SECTR, SECSY, MAXCY, POLSY, REFOR, RESCY, DISPN
034721  %           SECWO = WORDS   / SECTOR.
034721  %           SECTR = SECTORS / TRACK.
034721  %           SECSY = SECTORS / CYLINDER.
034721  %           MAXCY = VALUE OF MAX CYLINDER.
034721  %           POLSY = VALUE OF FIRST CYLINDER IN POOL.
034721  %           REFOR = FORMAT TYPE:
034721  %                       0 = TRACK OR NO REALLOCATION.
034721  %                      20 = SECTOR REALLOCATION.
034721  %           RESCY = VALUE OF FIRST RESERVED CYLINDER.
034721  %           ALTFO = ALTERNATIVE FORMAT (ADDRESS OF DTxxx OR 0)
034721  %           DISPN = INDEX FOR THIS ENTRY(FORMAT).
034721  %
034721
034721  "88D1+88D2+88D3+88D4
034721
034721  DT030, 1000; 22;   22; 1466; 1465;  0;    0;      0; 20 % DISC-30/60/90MB
034732  DT037, 1000; 22;  132;  632;    0;  0;    0;      0; 21 % DISC-38MB
034743  DT070, 1000; 22;  132; 1466; 1434;  0; 1465;      0; 22 % DISC-70MB
034754  DT075, 1000; 22;  132; 1466; 1465;  0;    0;      0; 23 % DISC-75MB
034765  DT140, 1000; 22;  264; 1466; 1441;  0; 1465; DT135; 24 % DISC-140MB-F
034776  DT135, 1000; 21;  252; 1466; 1462; 20; 1465; DT140; 25 % DISC-140MB-S
035007  DT160, 1000; 22;  264; 1466; 1465;  0;    0;      0; 26 % DISC-2-75MB
035020  DT288, 1000; 22;  526; 1466; 1465;  0;    0;      0; 27 % DISC-288MB-R
035031  DT285, 1000; 21;  630; 1306; 1303; 20; 1305; DT300; 30 % DISC-288MB-S
035042  DT300, 1000; 22;  660; 1306; 1260;  0; 1305; DT285; 31 % DISC-288MB-F
035053  DT450, 1000; 32; 1160; 1306; 1303; 20; 1305; DT460; 32 % DISC-450MB-S
035064  DT460, 1000; 33; 1210; 1306; 1260;  0; 1305; DT450; 33 % DISC-450MB-F
035075  DTXXX, 1000; 22; 1254; 1466;    0;  0; 1465;      0; 34 % SPARE (600 MB)
035106  DTYYY, 1000; 22; 1254; 1466;    0;  0; 1465;      0; 35 % SPARE (600 MB)
035117  DTZZZ, 1000; 22; 1254; 1466;    0;  0; 1465;      0; 36 % SPARE (600 MB)
035130
035130  "
035130  %       END LAYOUT
035130
035130  DLEZ=11                         % DISC LAYOUT TABLE ELEMENT SIZE
035130
035130  %YENDC=*; )KILL 9POFS; 9POFS=YENDC; )KILL 9ENDC
035130  %XENDC/; )KILL XENDC
```

```
035130
035130  "7B1U0;              XXUN0=DT075        % CONTROLLER 1; UNIT 0: 75MB
035130  "8BD1
035130  XXXUN=0
035130  "8BD1
035130  % BIG DISC DATAFIELD
035130          0;0
035132          0;0;0;0;0;0;0;0;0;0
035144          0;0;0;0;0;0;360;0
035154          0;-4;-34;0;0;0;0;0;0
035165          0;0;0;0;0;0;0;0;0;0;0
035200          0;0;20;0;-4;0
035206          0;1000;BDISK;BUSYE;COOPT;PFEIL
035214          MTMRS;0;-4;1540;CTRDI;0
035222  BIGDI,  0;0;*-2;2;0;0;RETRA;0
035232          0;MTRNS;0;0;0;0;0;0
035242          XXXUN;XXUN0;XXUN1;XXUN2;XXUN3
035247  )KILL XXUN0 XXUN1 XXUN2 XXUN3 XXXUN
035247          0
035250          0;DEDF1
035252
035252
035252  "8DILG
035252
035252  % DISC ACCESS LOG DATAFIELD
035252
035252          DVDIL;DVDIL
035254  DFDIL,  0;0;*-2;2;0;0;RETRA;0
035264          0;MTRNS;0;0;0;0;0;0
035274          0;0;0;0;0;0;0;0
035304          0;0;0;0;0;0;0;0
035314          0;0;0;0;0;0
035322  DALM1,  *-5;*-5;*-4;*+1; 1      % PARAMETER LIST FOR MON ABSTR  WHEN WRITING
035327                                  % TO DISC LOG FILE
035327  XP131,  *+2;*+2                 % PARAMETER LIST FOR MON ABSTR WHEN UPDATING
035331                                  % DISC ACCESS LOG DATAFIELD
035331  XDILF,  0;0;DILBU
035334  DILBU=*;  *+16/
035352
035352  XDILD,  0;MLDIL                 % DATAFIELD FOR LINKING INTO MONITOR QUEUE
035354  "99SM1
035354
035354  % STC MAG.TAPE DATAFIELD
035354          0;0;0;0;0;0
035362          0;0;0;0;0;0;0;0
035372          0;89NMT;0;0;2;0;4;0
035402          0;0;0;0;0;0;0;0
035412          0;1610;0;0;-4;0
035420          -2;2000;SMAGT;MBUSY;MFINI;MFEIL
035426          MTMRS;0;-2;520;CTRMA;0
035434  MTFIE,  0;0;*-2;1006;0;0;RETRA;0
035444          0;MTRNS;0;0;0;0;0;0
035454          MTCLD;6100;0;0;0;0;0;0
035464          0;0;0;0;0;0;0;0
035474          0;0;0;0;0;0;0;0
035504          0;0;0;0;0;0;0;0;MTDI1
035514          MTDI2;MTDI3;MTDI4;0;0
035521          0;0;0;0;0;0;0;0
035531
035531  )FILL
```

```
035531
035531  "99SM1 8M1U0
035531         STMRS;0;-2;0;0
035536  "8M1U0
035536
035536         0
035537  MTDI1, 0;0;*-2;112000;0;0;IORES
035546         CBGET;TEXIT;CAICL;MTDO1;0
035553         0;4000;0;0;4000;0
035561         560;526;0;0;0;0;0
035570         (0;*+3;*-6;*-4;0;0;0;40
035600
035600         0
035601  MTDO1, 0;0;*-2;112000;0;0;IORES
035610         CBPUT;TEXIT;CAOCL;MTDI1;0
035615         0;4000;0;0;4000;0
035623         560;526;0;0;0;0;2000
035632         (1;*+3;*-6;*-4;0;0;0;40
035642
035642  )FILL
035644
035644  "8BFD1
035644  % NEW FLOPPY DISC 1 DATAFIELD
035644
035644         0                                  % CFLUN
035645         0                                  % USED BY NEW FLOPPY/STREAMER
035646         0;0;0;0                            % DOORL(4)
035652         1;1;1;1                            % STEPR(4) (FOR NEW DRIVE, NEW CONTR., 0)
035656         0;0;0;0                            % PRECP(4)
035662         0;0;0;0                            % DTRCK(4)
035666         0;0;0;0;0;0;0;0;0;0;0;0;0;0        % FCOMF(14)
035702
035702         0;0;0;0;0;0                        % FRETURN,FDIMOD,FDIFORM(4)
035710         1150;2203;3722;  -1                % LFADDR(20)    FORMAT  0- 3
035714         2320;4406;7644;  -1                %                       4- 7
035720         2203;3722;  -1;1150                %                      10-13
035724         4406;7644;  -1;2320                %                      14-17
035730          400; 200; 100;  -1                % WDSCT(20)
035734          400; 200; 100;  -1
035740          400; 200;  -1;1000
035744          400; 200;  -1;1000
035750         0;0;0;0;0;0;0;0                    % NFDIADR(4),SHSTAT(4)
035760         0;99NFD                            % CERRCODE,MAXUNIT
035762         0;0;0;0;0;0                        % UNUSED,UNUSED,UNUSED,BUSFL,UNUSED,CFLRG
035770         0;0;0;0;0;0;0;0                    % TRG,ARG,DRG,XRG,CTRG,CARG,CDRG,CXRG
036000         0;0;0;0;-12;0                      % ERCNT,SERRB,WERRB,AERRB,TACNS,TACOUNT
036006         0;0;BFDIS;FDIBU;FDIFI;FDIFE        % COMFL,BLSZ,TRNSF,BUSY,FINISH,ERROR
036014         FDTMS;0;-10;IOX 1560;TFDIS;0       % TMSUB,TMR,TTMR,HDEV,STDRIV,DRIVER
036022  FDID1, 0;0;*-2;402;0;0;RETRA              % RESLINK,RTRES,BWLINK,TYPRING,ISTATE,MLINK,MFUNC
036031         0;0;MTRNS;0;0                      % TRLREG,HSTAT,MTRANS,UNUSED,UNUSED
036036         0;0;0;0;0                          % MEMA1,MEMA2,CMAD1,CMAD2,CLEDEV
036043
036043  "8F1U0
036043
036043         0
036044  F1U0I, 0;0;*-2;112000;0;0;IORES
036053         CBGET;TEXIT;CAICL;F1U00;0
036060         0;0;0;0;0;0
036066         1145;1146;0;0;0;0;0
036075         (0;*+3;*+6;*-4;0;0;0;1000;0
```

```
036106
036106          0
036107  F1U00,  0;0;*-2;112000;0;0;IORES
036116          CBPUT;TEXIT;CAOCL;F1U0I;0
036123          0;0;0;0;0;0
036131          1145;1146;0;0;0;0;0
036140          (1;*+3;*+6;*-4;0;0;0;1000;0
036151  )FILL
036153  .
036153  %=============================================================================
036153  %               D A T A F I E L D S   F O R   E X A B S T R
036153  %
036153  FPL3A=*
036153          0;0;*-2;2;0;0;0;0
036163          0;0;0;0;0;0;0;0
036173          0;0;0;0;0;*-21;*-21;*-7;*-6
036204
036204          0;0;*-2;2;0;0;0;0
036214          0;0;0;0;0;0;0;0
036224          0;0;0;0;0;*-21;*-21;*-7;*-6
036235  LPL3A=*
036235          0;0;*-2;2
036241  %=============================================================================
036241
036241
036241
```

```
036241
036241
036241  %=============================================================================
036241  % 28.5        T E R M I N A L  D A T A F I E L D S
036241  %
036241  % TERMINAL 1 INPUT DATAFIELD
036241
036241  NTRMS=0
036241  5TTST=*                          % USED BY ND-500 SYSTEM
036241
036241  "CXCPU
036241  ECBR7=1                          % BREAK AND ECHO TABLE #7 ALWAYS PRESENT
036241
036241  GTERM 01,300
036241          BAK01;300;0;1
036245  DT01R,  0;0;33;110040;-1;44005;IORES
036254
036254          0;300+4;0;0
036260  DT01W,  0;0;*-2;110040;0;0;IORES
036267  XYZ=NTRMS; )KILL NTRMS; NTRMS=XYZ+1; )KILL XYZ
036267
036267
036267  XXYYZ=*
036267  DT01W+TYPRI/^10                  % SET BIT 5NORESERVE
036264  XXYYZ/; )KILL XXYYZ
036267  "
036267  9CXTI=DT01W-DT01R                % USED BY INTDFIELDS
036267  5TTSZ=*-5TTST                    % USED BY ND-500 SYSTEM
036267
036267
036267  % TERMINAL 2 - 64 DATAFIELDS
036267
036267  "8TR5; GTERM 05,340
036267          BAK05;340;0;1
036273  DT05R,  0;0;33;110040;-1;44005;IORES
036302
036302          0;340+4;0;0
036306  DT05W,  0;0;*-2;110040;0;0;IORES
036315  XYZ=NTRMS; )KILL NTRMS; NTRMS=XYZ+1; )KILL XYZ
036315
036315  "8TR6; GTERM 06,350
036315          BAK06;350;0;1
036321  DT06R,  0;0;33;110040;-1;44005;IORES
036330
036330          0;350+4;0;0
036334  DT06W,  0;0;*-2;110040;0;0;IORES
036343  XYZ=NTRMS; )KILL NTRMS; NTRMS=XYZ+1; )KILL XYZ
036343
036343  "
036343
036343  % TERMINAL 64-28  DATAFIELDS
036343
036343
036343
036343  5TEND=*                          % USED BY ND-500 SYSTEM
036343
036343  "BAD01
```

```
036343
036343  %=============================================================================
036343  % 28.5B       B A C K G R O U N D   A C C E S S   D E V I C E
036343  %             D A T A F I E L D S
036343
036343  "BAD01;BADDV 01
036343
036343          0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
036364          0;-1;0;0;0;0;33;0;3;410;0;ECH0;BRK0;0;0;0;0;0;0;0
036410  BD01R,   0;0;*-2;112020;0;0;IORES;BDGET;TEXIT;CTIBA;BD01W;0
036424          0;0;0;0;0;0;0;0;TAD01;BFI01;0;0;0;0;0;0;0;0;0;0
036450          0;0;0;0;0;0;0;0
036460
036460          0;0;BDTOU;0;-3;0;0;0
036470  BD01W,  0;0;*-2;112020;0;0;IORES;BDPUT;BDOUT;CTOBA;BD01R;0
036504          0;0;0;0;0;0;0;0;0;0;0;0;0;BDROU;0;0;0
036524          0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
036543
036543  '
036543
036543
036543
036543  "8SM01+8SM02+8SM03+8SM04+8SM05+8SM06+8SM07+8SM08+8SM09+9SM01+9SM02+9SM03+9SM04+9SM05+9SM06+9SM07
036543  %=============================================================================
036543  % 28.8        M O D E M   D A T A F I E L D S
036543  %
036543  "8SM01
036543  % MODEM INPUT DATAFIELD
036543
036543          0;0;0;0;MOTMI;0;-16
036552          IOX 100;MODIN;MODI1
036555  IDM01,  0;0;*-2;110000;0;0;IOREM
036564          MOTRI;TEXIT;MOSTI;UDM01;0
036571          BUF;MIBU1+MIBU1;0;0;MIBU1+MIBU1;0;0
036600          MIBU1+MIBU1
036601
036601  %%XBU=BUF+MIBU1;)KILL BUF;BUF=XBU;)KILL XBU
036601
036601  "8SM01
036601  % MODEM OUTPUT DATAFIELD
036601
036601          0;0;0;0;MOTMO;0;-3
036610          IOX 104;MODUT;MODUT
036613  UDM01,  0;0;*-2;110000;0;0;IORES
036622          MOTRO;TEXIT;MOSTO;IDM01;0
036627          BUF;MOBU1+MOBU1;0;0;MOBU1+MOBU1;0
036635
036635  %%XBU=BUF+MOBU1;)KILL BUF;BUF=XBU;)KILL XBU
036635
036635  '
036635
```

```
036635
036635
036635  %=============================================================================
036635  % 28.10       L I N E   P R I N T E R   D A T A F I E L D S
036635  %
036635   8DLP1+8DVE1
036635
036635  % DATAFIELDS FOR LINE-PRINTER 1 USING MONITOR CALL MAGTP
036635
036635          0;1DVE1;0;COCHB;0;0;0;0;0;0;0;0;0;0
036653          400;0;LP1BU;0
036657          DMLP1;4;IOX 433;IOX 432;IOX 431;0;TRDLP
036666          0;0;0
036671          DLPTM;0;-3;IOX 430;TLPRI;CLP10
036677  DMLP1,  0;0;*-2;2006;0;0;RETRA
036706          0;0;MTRNS;0;DILP1;0;0;0;0;DMLPC;DILP1+37
036721  COCHB=*;  *+42/;  )KILL COCHB
036763  %%XBU=BUF+200;)KILL BUF;BUF=XBU;)KILL XBU
036763
036763          0
036764  DILP1,  0;0;*-2;112000;0;0;IORES
036773          CLPUT;TEXIT;CLCLO;0;0
037000          0;1000;0;0;1000;0
037006          1167;1170;0;0;0;0;0
037015          (1;*+3;*-6;*-4;0;0;11;5
037025          DMLP1
037026  )FILL
037027  )KILL 1DVE1
037027
037027
037027  "
037027
037027
037027
037027  %=============================================================================
037027  % 28.17       C O M M U N I C A T I O N   L I N E   D A T A F I E L D S
037027  %
037027
037027  MXLIN=*
037027  LINAR, ANLI
037030
037030  ANLI=*-LINAR-1
037030
037030  RPASS, 11;0;0;0;0;0;0;0;0;0
037042  FRSEG, 0
037043  AKMCS=0
037043  "BADAD
037043  CDVNR=602
037043  "
037043
037043
037043  %=============================================================================
037043  %  28.19   B A S E F I E L D S   F O R   R T   C O P Y   P R O G R A M S
037043  %
037044
037043
037043  LRIDT=*
037043
037043  "8MPM4
037043
```

```
037043  %=======================================================================
037043  % 28.22       DATAFIELD FOR MULTIPORT IV
037043  %
037043
037043          100200;BUSCD;BUSCD
037046  MPM4D, 0;0;*-2;2
037052  "
037052
037052
037052  )LINE
037052  %%%%%%%%%%%%  M A C R O E S  -  S I N 2  -  G E N  %%%%%%%%%%%%%%%%%%%
037052
037052  %%      MACROES USED ON SIN2-GEN
037052  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
037052  %%      PREVIOUSLY USED MACROES ARE KILLED
037052  )KILL DFFLD GTERM TELDF HDDMA HDIO CHIN CHUT CTLIN TTUT GXTRM
037052  )KILL VDDF VPDF VDMT
037052
037052  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
037052
037052
037052  % MACRO FOR INTERNAL DEVICE IN
037052
037052  )MCDEF IDVI $NAM,$NOC,$TCHAR
$NAM I,0;0;*-2;112100;0;0;IORES;IGTCH;ISTDV;CLBUF
        $NAM 0;0;BUF;$NOC +$NOC ;0;0;$NOC +$NOC ;0
        1210+XIBNX;1211+XIBNX;1;0;$STCHAR ;0
%%XBU=BUF+$NOC ;)KILL BUF;BUF=XBU;)KILL XBU
"8S3C
BUF=*; )KILL BUF; *+$NOC /
"
|
037052
037052  % MACRO FOR INTERNAL DEVICE OUT
037052
037052  )MCDEF IDVO $NAM
$NAM O.  0;0;*-2;112100;0;0;IORES;IPTCH;OSTDV;INIOS;$NAM I;0
       0;0;0;0;0;0
        1210+XIBNX;1272+YIBNY;0;0
YIBNX=XIBNX+2;)KILL XIBNX; XIBNX=YIBNX;)KILL YIBNX
YIBNZ=YIBNY+1;)KILL YIBNY; YIBNY=YIBNZ;)KILL YIBNZ
|
037052
037052  % MACRO FOR INTERNAL BLOCK DEVICE
037052
037052  )MCDEF BLIDV $NAM
       0;0;$NAM O
$NAM I, 0;0;*-2;102;0;0;0;0;0;IMTRI;0;0

        $NAM I
$NAM O, 0;0;*-2;102;0;0;0;0;0;IMTRO;0;0
|
037052
037052  % MACRO FOR SIBAS INT. DEV.:
037052  )MCDEF SIB $NAM
$NAM I,0;0;*-2;110000;0;0;IORES;GETDW;ISTDV;CLBUF
        $NAM 0;0;BUF;200;0;0;200;0;0;0;200;0;177;0
%%XBU=BUF+200;)KILL BUF;BUF=XBU;)KILL XBU

$NAM O, 0;0;*-2;110000;0;0;IORES;PUTDW;OSTDV;CEXIT;$NAM I;0
```

```
]
037052
037052
037052  % MACRO FOR ALLOCATING AREA WITH ZEROES
037052  )MCDEF BSPAC $NOADR
*<*+$NOADR -1
)ZERO
*+$NOADR /
]
037052  % MACRO FOR TEKTRONIX DISPLAY
037052
037052  )MCDEF TXTRO $NAM,$HNO,$MAXX,$MAXY,$MAXZ
TEXTR; $MAXY ;$MAXZ ;IOX $HNO ;0 ; $MAXX ;
$NAM . 0; 0; *-2; 0
]
037052
037052  % BATCH PROCESS MACRO
037052  )MCDEF BCHD $NAM,$GBACK,$SIZ

        33;0;0;0;0;0;0;0;0;0;0;0;0;0
$NAM R, 0;0;*-2;110002;0;0;IORES
        0;TEXIT;CEXIT;$NAM W;0
        0;0;0;0;BATNO;0
        0;0;$GBACK ;0;0;0
        0;0;0;0;0;0;0;0
        0;0;0

$NAM W, 0;0;*-2;110002;0;0;IORES
        0;0;CEXIT;$NAM R;0
        0;0;0;0;BATNO;0
        0;0;0

$NAM O, 0;0;*-2;110002;0;0;IORES
        IPTCH;OSTDV;CLBUF;$NAM I;0

$NAM I, 0;0;*-2;110002;0;0;IORES
        IGTCH;ISTDV;CLBUF;$NAM O;0
        BUF;$SIZ +$SIZ ;0;0;$SIZ +$SIZ ;0;0;0;20;0;47;0
%%XBU=BUF+$SIZ ;)KILL BUF;BUF=XBU;)KILL XBU
BNO=BATNO+1;)KILL BATNO;BATNO=BNO;)KILL BNO
]
037052
037052  %MACROES FOR NORDCOM (SGB,SEL,GB,TB)
037052
037052
037052
037052  "BUDMA+8VICO
037052  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
037052  %%       DATAFIELD DEFINITION AND
037052  %%       DATAFIELD INITIALIZATION FOR UDMA
037052  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
037052  )MCDEF GUDMA $NR,$HD

 %------------------------------------- UDMA (INPUT) DATAFIELD $NR ;
         0;0;0;0;0;
         0;0;0;0;0;0;0;0;
         0;0;0;0;0;0;0;0;
         0;0;0;0;0;0;ND852;UBUSY;
         UFIN;UBUSY;UDTMO; 0;177771;$HD ;USDRV;UDDRV;
UDI$NR .  0;0;*-2;2000;0;0;RETRA;0;
```

```
        0;MTRNS;0;0;0;0;
 %------------------------------------- OUPUT DATAFIELD FOR UDMA$NR ;
UDO$NR ,  0;0;*-2;0;0;0;0;*+4
        *+4;*+5;*+5;0;0;0;0;0          % ABSTR PARAMETER LIST
        0;UDF$NR ;*+1;0;0;0;0;0          % 22/*+1 DFPNT  MONITOR CALL
        0;0;0;0;0;0;0;0                %              WORKING AREA
        0;0;0;0;0;0;0;0                %
        0;0;0;0;0;0;0;0;0              % MAX DISP: 36
"8VICO+8UDMA 8F5UD
%------------------------------------- DATAFIELD FOR UDR$NR ;
UDF$NR .  0;0;*-2;0;0;0;RIOWA;0          % FAST UDMA FORM ND-500
        *+4;*+4;*+5;*+5;0;0;0;0        % ABSTR PARAMETER LIST
        0;0
"8UDMA+8VICO
)FILL
]
037052  "
037052
037052  "8GPI0+8GPI1+8GPI2+8GPI3+8GPI4+8GPI5+8GPI6+8GPI7
037052  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
037052  %%       DATAFIELD DEFINITION AND
037052  %%       DATAFIELD INITIALIZATION FOR GPIB
037052  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
037052  )MCDEF GGPIB $NO,$HDV
%--------------------------------------  GPIB (INPUT)DATAFIELD $NO ;
      GPIBT;4000;400;0;0;              % GPXTR DSIZE USIZE GPRUN GPBSI
      GPIBD;GPBUS;GPFIN;GPERR;         % STANDARD PART
      MTMRS;0;-7;$HDV ;GPITR;GPITR
DTGP$NO , 0;0;*-2;0;0;0;RETRANS;0;0;MTRNS
      #0$NO ;0;0;0;0;0;0;0               % GPIBN ACTIV XTBLK SXTBL CPORT SPORT DPORT DMESA
      0;0;0;0;0;0;0;0                  % DBUFA       MVERS SUSFL DEVFL USCON CGPIM CURMTY
      0;0;0;0;0;0;0;0                  % CURMN       ERADD XERCO SRQFL MRETA SRETA GRETA
      0;0;0;0;0;0;0;0                  % STREG SAREG SDREG SXREG DINPT PIODP PIOWC FIRST
      0;*+20/;*+40/                    % SRSTR UMESS(20)   MNARR(2*20)
      0;0;*+10/;0;0;0;*+40/            % CURFU CGPUS INSTA(10)   CURAD      CURBC UENV(2*20)
]
037052  "
037052
```

```
037052  )LINE%
037052  %=============================================================================
037052  %       S I N 2 - G E N
037052  %=============================================================================
037052
037052
037052
037052
037052
037052
037052
037052  %
037052
037052
037052  %=============================================================================
037052  % 28.42       S E M A P H O R E  D A T A F I E L D S
037052  %
037052  % SEMAPHORES
037052
037052  SEMI1, 0;0;*-2;0
037056  SEMI2, 0;0;*-2;0
037062  SEMI3, 0;0;*-2;0
037066  SEMI4, 0;0;*-2;0
037072  SEMI5, 0;0;*-2;0
037076
037076
037076  % SEMAPHORES RESERVED FOR THE BACKUP-SYSTEM
037076  SEM60, 0;0;*-2;2
037102  SEM61, 0;0;*-2;2
037106  SEM62, 0;0;*-2;2
037112  SEM63, 0;0;*-2;2
037116  SEM64, 0;0;*-2;2
037122
037122
```

```
037122
037122  %=============================================================================
037122  % 28.43       F I L E S Y S T E M  S E M A P H O R E  D A T A F I E L D S
037122  %
037122  % FILE SYSTEM SEMAPHORES
037122
037122
037122  9SFIS=*
037122  REESM, 0;0;*-2;2
037126  RTLFI, 0;0;*-2;2
037132  FS4,   0;0;*-2;2
037136  FS5,   0;0;*-2;2
037142  FS6,   0;0;*-2;2
037146  FS7,   0;0;*-2;2
037152
037152  FS20,  0;0;*-2;0              % GRAFS SEMAPHORE NO. 2
037156  FS21,  0;0;*-2;2
037162
037162  FS27,  0;0;*-2;2
037166  "8MT1+8MT2+8MT3+8MT4
037166  FS61,  0;0;*-2;2
037172  "3B1U0+6B1U0+7B1U0+2B1U0+4B1U0+3C1U0+6C1U0+9C1U0+3D1U0+2D1U0+1E1U0+2E1U0+3E1U0+5E1U0+7E1U0+8E1U0+EE1U0+FE1U
037172  F1101, 0;0;*-2;2
037176  F1102, 0;0;*-2;2
037202  "SLP1+CSP12
037202  F1136, 0;0;*-2;2
037206  F1137, 0;0;*-2;2
037212  "
037212  F1142, 0;0;*-2;2
037216  F1143, 0;0;*-2;2
037222  F1144, 0;0;*-2;2
037226  % ************
037226  "8F1U0
037226  F1150, 0;0;*-2;2
037232  F1151, 0;0;*-2;2
037236  "
037236  F1177, 0;0;*-2;2
037242  "-8S3C
037242  D1201, 0;0;*-2;2              % DMAC
037246  "8LOG
037246  F1202, 0;0;*-2;2              % RT-PROGRAM-LOG
037252  "
037252  F1204, 0;0;*-2;2              % SERVICE PROGRAM
037256  F1205, 0;0;*-2;2              % MAIL
037262  F1222, 0;0;*-2;2              % TERMINATION COMMANDS BUFFER LOCK
037266  F1223, 0;0;*-2;0              % GRAFS SEMAPHORE NO. 1
037272  "8COSP
037272  F1731, 0;0;*-2;2                        % COSMOS SPOOLING PERIPHERAL DEVICE
037276  F2166, 0;0;*-2;2                        % COSMOS SPOOLING QUEUE SEMAPHORE
037302  F2167, 0;0;*-2;2                        % COSMOS SPOOLING I/O SEMAPHORE
037306  "
037306
037306  "8NSDB
037306  % WORKING DATAFIELD FOR DEBUGGER MON CALL
037306  % USED TO ENSURE CORRECT SYNCRONIZATION
037306  DDEBU, 0;0;*-2;0;0;0;0;
037315         0;0;0   ;0;0;0;0;
037324         0;0;0   ;0;0;0
037332
037332  "
```

```
037332  UESEM, 0;0;*-2;2          % UEADM SEMAPHORE
037336  NAMSE, 0;0;*-2;2          % RTFIL SEMAPHORE
037342  ACSEM, 0;0;*-2;0          % ACCOUNTING SEMAPHORE
037346  "8DILG
037346  SEMDL, 0;0;*-2;2          % DISC-ACCESS-LOG BUFFER LOCK
037352  "
037352
037352  9EFIS=*
037352
037352  F1200, 0;0;*-2;0          % *** SPECIAL FOR NORD TPS ***
037356  "8LOG
037356  F1352, 0;0;*-2;2
037362  "BADAD
037362  BDSEM, 0;0;*-2;2
037366  BASEM, 0;0;*-2;2
037372  "
```

```
037372
037372  %=============================================================================
037372  % 28.44       I N T E R N A L   D E V I C E   D A T A F I E L D S
037372  %
037372  % INTERNAL DEVICES
037372
037372  "IND01+IBL01
037372  IDVI ID01,IDBUS,TERMC
037372
037372  ID01I,0;0;*-2;112100;0;0;IORES;IGTCH;ISTDV;CLBUF
037404          ID010;0;BUF;IDBUS+IDBUS;0;0;IDBUS+IDBUS;0
037414          1210+XIBNX;1211+XIBNX;1;0;TERMC;0
037422  %%XBU=BUF+IDBUS;)KILL BUF;BUF=XBU;)KILL XBU
037422  "
037422
037422
037422  IDVO ID01
037422
037422  ID010,  0;0;*-2;112100;0;0;IORES;IPTCH;OSTDV;INIOS;ID01I;0
037436          0;0;0;0;0;0
037444          1210+XIBNX;1272+YIBNY;0;0
037450  YIBNX=XIBNX+2;)KILL XIBNX; XIBNX=YIBNX;)KILL YIBNX
037450  YIBNZ=YIBNY+1;)KILL YIBNY; YIBNY=YIBNZ;)KILL YIBNZ
037450
037450
037450  "IBL01
037450  BLIDV IB01
037450          0;0;IB010
037453  IB01I, 0;0;*-2;102;0;0;0;0;0;IMTRI;0;0
037467
037467          IB01I
037470  IB010, 0;0;*-2;102;0;0;0;0;0;IMTRO;0;0
037504
037504
037504  "IND02+IBL02
037504  IDVI ID02,IDBUS,TERMC
037504
037504  ID02I,0;0;*-2;112100;0;0;IORES;IGTCH;ISTDV;CLBUF
037516          ID020;0;BUF;IDBUS+IDBUS;0;0;IDBUS+IDBUS;0
037526          1210+XIBNX;1211+XIBNX;1;0;TERMC;0
037534  %%XBU=BUF+IDBUS;)KILL BUF;BUF=XBU;)KILL XBU
037534  "
037534
037534
037534  IDVO ID02
037534
037534  ID020,  0;0;*-2;112100;0;0;IORES;IPTCH;OSTDV;INIOS;ID02I;0
037550          0;0;0;0;0;0
037556          1210+XIBNX;1272+YIBNY;0;0
037562  YIBNX=XIBNX+2;)KILL XIBNX; XIBNX=YIBNX;)KILL YIBNX
037562  YIBNZ=YIBNY+1;)KILL YIBNY; YIBNY=YIBNZ;)KILL YIBNZ
037562
037562
037562
037562  "BHIST
037562  H1203, 0;0;*-2;2              % HISTOGRAM SEMAPHORE
037566  "
037566
037566
037566  %=============================================================================
```

```
037566  % 28.45       DATAFIELD FOR USERS RESTART PROGRAM
037566
037566  URERT, 0;0;0;0;0;0;0;0;0;DMONI
037600
037600  "-8S3C
037600
037600  %-------------------------------------------------------------------------
037600  % 28.46       ABSTRANS PARAMETERS FOR START-PROGRAM
037600
037600  SBANK, 0                        % SBUF MUST BE A DOUBLE WORD
037601  SBUF,  0
037602  DFBLC=*
037602  FBLCK, 0;0
037604  DTBLC=*
037604  TBLCK, 0;0
037606  XFUNC, 0
037607  NBLCK, 0
037610  RPAR,  XFUNC
037611         SBUF-1
037612         FBLCK
037613         NBLCK
037614  WPAR,  XFUNC
037615         SBUF-1
037616         TBLCK
037617         NBLCK
037620
037620  )FILL
037620  "
037620
037620  %=========================================================================
037620  % 28.47       B A T C H  D A T A F I E L D S
037620  %
037620  BATNO=1
037620
037620  "8BCH1
037620  BCHD BT01,BCH01,300
037620
037620         33;0;0;0;0;0;0;0;0;0;0;0;0;0
037636  BT01R, 0;0;*-2;110002;0;0;IORES
037645         0;TEXIT;CEXIT;BT01W;0
037652         0;0;0;0;BATNO;0
037660         0;0;BCH01;0;0;0
037666         0;0;0;0;0;0;0;0
037676         0;0;0
037701
037701  BT01W, 0;0;*-2;110002;0;0;IORES
037710         0;0;CEXIT;BT01R;0
037715         0;0;0;0;BATNO;0
037723         0;0;0
037726
037726  BT010, 0;0;*-2;110002;0;0;IORES
037735         IPTCH;OSTDV;CLBUF;BT01I;0
037742
037742  BT01I, 0;0;*-2;110002;0;0;IORES
037751         IGTCH;ISTDV;CLBUF;BT010;0
037756         BUF;300+300;0;0;300+300;0;0;0;20;0;47;0
037772  %%XBU=BUF+300;)KILL BUF;BUF=XBU;)KILL XBU
037772  BNO=BATNO+1;)KILL BATNO;BATNO=BNO;)KILL BNO
037772
037772
```

```
037772  "
037772
037772
037772
037772
037772
037772
037772
037772  %=============================================================================
037772  % 28.51      A N A L O G U E  I N P U T  D A T A F I E L D S
037772  %
037772  %
037772  % "CONNECT"-DATAFIELDS FOR ANALOGUE INPUT (ND-820).
037772  %
037772  "8(XHD
```

```
037772
037772  %=============================================================================
037772  % 28.52     G E N E R A L   H D L C  -  D M A  -  D A T A F I E L D S
037772  %
037772  ENSIZ=1520
037772
037772
037772  "8C1HD
037772  %
037772  %       HDLC OUTPUT-DATAFIELD 1
037772  %
037772          X21OP;POFTO;0;HDTM2;0;-2;IOX 1640;HDSTA;FALSE
040003  HDOF1,   0;0;*-2;0;0;0;IORES
040012          0;ACT12;0;HDIF1;0
040017          0;0;0;0;0;0
040025          0;0;0;0;0;0;0
040034          0;0;0;0;0;0;0
040043          0;0;0;0;0;0;0;0;0;0;0
040056          0;0;0;0;0;0;0;0;0;
040067          HXCOD;0;0;0;0;WT12;TRASE;XSDAT;77          % *81F*
040100          0;0;HOLI1;0;0;0;0
040107          0;0;0;*+1;0;0;0;0
040117          0;0;0;0
040123          0
040124
040124  HOLI1,   0;0;0;0
040130          NLP;0;0;HOLI1
040134  %
040134  %       HDLC INPUT-DATAFIELD 1
040134  %
040134          X21OP;HIINT;0;HDTM3;0;-2;IOX 1640;HDSTA;HIINT
040145  HDIF1,  0;0;*-2;0;0;0;IORES
040154          0;ACT13;0;HDOF1;0
040161          0;0;0;0;0;0
040167          0;0;0;0;0;0;0
040176          0;0;0;0;0;0;0
040205          0;0;0;0;0;0;0;0;0;0;0
040220          0;0;0;0;0;0;0;0;0;0
040231          HXCOD;0;0;0;0;WT13;RECSE;HDREC;0
040242          0;0;HILI1;0;0;0;0
040251          0;0;0;*+1;0;0;0;0
040261          0;0;0;10
040265          0;0;0;0;0;0;0;0
040275  HILI1,   0;0;0;0
040301          0;0;0;0
040305          0;0;0;0
040311          0;0;0;0
040315          0;0;0;0
040321          0;0;0;0
040325          0;0;0;0
040331          0;0;0;0
040335          NLP;0;0;HILI1
040341
040341
040341  "8C2HD
040341  %
040341  %       HDLC OUTPUT-DATAFIELD 2
040341  %
040341          X21OP;POFTO;0;HDTM2;0;-2;IOX 1660;HDSTA;FALSE
040352  HDOF2,   0;0;*-2;0;0;0;IORES
```

```
040361          0;ACT12;0;HDIF2;0
040366          0;0;0;0;0;0
040374          0;0;0;0;0;0;0
040403          0;0;0;0;0;0;0
040412          0;0;0;0;0;0;0;0;0;0;0
040425          0;0;0;0;0;0;0;0;0
040436          HXCOD;0;0;0;0;WT12;TRASE;XSDAT;77          % *81F*
040447          0;0;HOLI2;0;0;0;0
040456          0;0;0;*+1;0;0;0;0
040466          0;0;0;0
040472          0
040473
040473  HOLI2,   0;0;0;0
040477          NLP;0;0;HOLI2
040503  %
040503  %       HDLC INPUT-DATAFIELD 2
040503  %
040503          X21OP;HIINT;0;HDTM3;0;-2;IOX 1660;HDSTA;HIINT
040514  HDIF2,  0;0;*-2;0;0;0;IORES
040523          0;ACT13;0;HDOF2;0
040530          0;0;0;0;0;0;0
040536          0;0;0;0;0;0;0
040545          0;0;0;0;0;0;0
040554          0;0;0;0;0;0;0;0;0;0;0
040567          0;0;0;0;0;0;0;0;0
040600          HXCOD;0;0;0;0;WT13;RECSE;HDREC;0
040611          0;0;HILI2;0;0;0;0
040620          0;0;0;*+1;0;0;0;0
040630          0;0;0;10
040634          0;0;0;0;0;0;0;0
040644  HILI2,   0;0;0;0
040650          0;0;0;0
040654          0;0;0;0
040660          0;0;0;0
040664          0;0;0;0
040670          0;0;0;0
040674          0;0;0;0
040700          0;0;0;0
040704          NLP;0;0;HILI2
040710
040710
040710  "8PIOC
040710  %=========================================
040710  % 28.53      P I O C  -  D A T A F I E L D S
040710
040710  %       PIOC SEMAPHORE (PIOC DEMFIELD)
040710
040710  SEMPI,  0;0;*-2;0;0;0;0                                  % 6
040717          0;0;0;0                                          % 10
040723          0;0;0;0;0;0;0;0;0;0                              % 20
040735          0;0;0;0;0;0;0;0;0;0                              % 30
040747
040747
040747  "8PC01
040747          PITIM;0;-2;140020;PDRIV;PDRIV
040755  PIO01,  0;0;*-2;0;0;0;PIORE                              % 6
040764          60;0;0;173                                       % 10
040770          0;0;0;0;0;0;0;0;0;0;0;0;77;0                     % 23
041006          0;0;0;0;0;0;0;0
041016          0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0                  % XT BLOCKS
```

```
041036
041036
041036
041036
041036
041036   8C1X2+8C2X2+8C3X2+8C4X2+8C5X2+8C6X2+8C7X2+8C8X2
041036  %=======================================
041036  % 28.54       X 2 1    D A T A F I E L D S
041036
041036  "8C1X2
041036           X321T;0;HDTM3;0;0;IOX 1640;X32ST;X321I
041046  X21F1,   0;0;*-2;0;0;0;IORES
041055           0;ACT13;0;0;0
041062           0;0;0;0;0;0
041070  %        FROM DEMFIELD
041070
041070           0;0;0;0;0;0;0
041077           0;0;0;0;0;0;0
041106           0;0;0;0;0;0;0;0;0;0;0
041121           0;0;0;0;0;0;0;0;0
041132  %        ONLY X21 USED
041132           HXCOD;0;0;0;0;0;0;0;0;0;0;0;0
041147  %   X2DPS       CALL PROGRESS SIGNALS, LINE ID, CHARGING INFO
041147           0;*+1;0;*+1;0;0;0;0;0;0
041161  %   X2DMP       MESSAGE POINTER
041161           0;0;0;0;0;0
041167  %   X2D00       HARDWARE REGISTERS
041167           0;0;0;0;0;0;0
041176  %   X2DL3
041176           0;0;0;0;0;0;0
041205  "
041205  %
041205
041205
041205  %
041205
041205  %
041205
041205
041205  %
041205
041205
041205
041205
041205
041205   "8UDMA+8VICO
```

```
041205
041205  %=============================================================================
041205  %             DATAFIELDS FOR UNIVERSAL DMA INTERFACES (INCLUDING VICOM)
041205  %
041205  "8UD01; GUDMA 01,140050
041205
041205   %---------------------------------------- UDMA (INPUT) DATAFIELD 01;
041205           0;0;0;0;0;
041212           0;0;0;0;0;0;0;0;0;
041222           0;0;0;0;0;0;0;0;0;
041232           0;0;0;0;0;0;ND852;UBUSY;
041242           UFIN;UBUSY;UDTMO; 0;177771;140050;USDRV;UDDRV;
041252  UDI01,   0;0;*-2;2000;0;0;RETRA;0;
041262           0;MTRNS;0;0;0;0;
041270   %---------------------------------------- OUPUT DATAFIELD FOR UDMA01;
041270  UDO01,   0;0;*-2;0;0;0;0;*+4
041300           *+4;*+5;*+5;0;0;0;0;0                 % ABSTR PARAMETER LIST
041310           0;UDF01;*+1;0;0;0;0;0                 % 22/*+1 DFPNT  MONITOR CALL
041320           0;0;0;0;0;0;0;0                       %               WORKING AREA
041330           0;0;0;0;0;0;0;0                       %
041340           0;0;0;0;0;0;0;0;0                     % MAX DISP: 36
041351  "8VICO+8UDMA 8F5UD
041351  %---------------------------------------- DATAFIELD FOR UDR01;
041351  UDF01,   0;0;*-2;0;0;0;RIOWA;0                 % FAST UDMA FORM ND-500
041361           *+4;*+4;*+5;*+5;0;0;0;0               % ABSTR PARAMETER LIST
041371           0;0
041373  "8UDMA+8VICO
041373  )FILL
041373
041373  "
041373  "8GPI0+8GPI1+8GPI2+8GPI3+8GPI4+8GPI5+8GPI6+8GPI7
```

```
041373
041373  %=============================================================================
041373  %              DATAFIELDS FOR GPIB INTERFACES
041373  %
041373  !8GPIO; GGPIB 0,140000
041373  %!-------------------------------------- GPIB (INPUT)DATAFIELD 0;
041373          GPIBT;4000;400;0;0;                % GPXTR DSIZE USIZE GPRUN GPBSI
041400          GPIBD;GPBUS;GPFIN;GPERR;           % STANDARD PART
041404          MTMRS;0;-7;140000;GPITR;GPITR
041412  DIGP0,  0;0;*-2;0;0;0;RETRANS;0;0;MTRNS
041424          #00;0;0;0;0;0;0;0                % GPIBN ACTIV XTBLK SXTBL CPORT SPORT DPORT DMESA
041434          0;0;0;0;0;0;0;0                   % DBUFA        MVERS SUSFL DEVFL USCON CGPIM CURMTY
041444          0;0;0;0;0;0;0;0                   % CURMN        ERADD XERCO SRQFL MRETA SRETA GRETA
041454          0;0;0;0;0;0;0;0                   % STREG SAREG SDREG SXREG DINPT PIODP PIOWC FIRST
041464          0;*+20/;*+40/                     % SRSTR UMESS(20)   MNARR(2*20)
041545          0;0;*+10/;0;0;0;*+40/             % CURFU CGPUS INSTA(10)   CURAD       CURBC UENV(2*20)
041622
041622
041622
041622
041622
041622
041622  !OCT00+OCT1+OCT2+OCT3
```

```
041622  %
041622  %=============================================================================
041622  %   DATAFIELS FOR OCTOBUS
041622  % octobus unit table
041622  OCTOP,OCTOD
041623  "
041623
041623
041623
041623
041623  )LINE,OCTOD

041623  %%%%%%%%%%%  M A C R O E S  -  S I N 3  -  G E N  %%%%%%%%%%%%%%%%%%%
041623
041623  %%      MACROES USED ON SIN3-GEN
041623  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
041623  %%      PREVIOUSLY USED MACROES ARE KILLED
041623  )KILL IDVI IDVO BLIDV SIB TXTRO BCHD NSGB NTBA DR11C DIGOUT
041623  )KILL ARISTO CMAC GUDMA GGPIB
041623
041623  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
041623
041623
041623
041623  % MACRO FOR BACKGROUND PROGRAMS
041623
041623  "-8S3C
041623
041623  )MCDEF BBP $NAM
$NAM .  0;1100;0;0;0;0;9ENTO;GSEG
       0;0;100000;*-13
       0;0;0
XYS=GSEG;)KILL GSEG;GSEG=XYS+1000;)KILL XYS
]
041623
041623
041623  "
041623
041623  )MCDEF JONAC $CJOC
CKJU=*; DVTAB/ $CJOC  %
CKJU/
)KILL CKJU
]
041623
041623
041623  % MACRO FOR REMOTE TERMINAL OUTPUT RT-DESCRIPTION
041623
041623  )MCDEF RTUP $NR,$LNR
RU$NR $LNR , 0;1061;0;0;0;0;RTOCH;0
       0;0;0;*-13
       0;0;0
]
041623
041623
041623  % MACRO FOR RWRT-PROGRAM FOR SIN/SIN CHANNEL INPUT
041623
041623  )MCDEF RWCR $NR,$LNR
CR$NR $LNR , 0;1100;0;0;0;0;TR$NR $LNR ;6
       0;0;0;*-13
       0;0;0
```

```
]
041623
041623   % MACRO FOR RWRT-PROGRAM FOR SIN/SIN CHANNEL OUTPUT
041623
041623   )MACDEF RWCW $NR,$LNR
CW$NR $LNR , 0;1100;0;0;0;0;TW$NR $LNR ;6
        0;0;0;*-13
        0;0;0

]
041623
041623   )MACDEF ACRTP
        NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
]
041623
041623
041623   )MACDEF GUDRT $NO
041623   % MACRO  UDMA RT-PROG   (FAST UDMA FROM ND-500)
041623   )MACDEF GUDRT $NO
UDR$NO , 0;1100;0;0;0;0;URT$NO ;0
        0;0;017612;UDR$NO ;0;0;0              % RING 2 ,PT=3,ATP=3
)FILL
]
041623
041623
041623
```

```
041623  )LINE
041623  %%%%%%%%%%%%%%%%%%%%%%% S I N 3  -  G E N %%%%%%%%%%%%%%%%%%%%%%%%%%
041623
041623  "-8CLI1 -8CLI2 -8CLI3 -8CLI4
041623  BRECC, EXIT
041624  "
041624
041624
041624
041624  %=============================================================================
041624  % 29.2        B A C K G R O U N D  T A B L E
041624  %
041624  % BACKGROUND TERMINAL TABLE
041624  1NBPP,0       % NUMBER OF BACKGROUND PROGRAMS (NOT EQUAL ENTRIES IN BACKGROUND TABLE)
041625  2NBPP,0       % NUMBER OF DYNAMIC ALLOCATED BACKGROUND PROGRAMS
041626  BNUMB, NOBGP
041627  BACKT,DT01R
041630  "8TR5 8BACS+8BP5;DT05R
041631  "8TR6 8BACS+8BP6;DT06R
041632  "BAD01;BD01R
041633  "
041633  BBCHT=*
041633  "8BCH1;BT01R
041634  "
041634  EBCHT=*
041634
041634
041634  NOBGP=*-BACKT
041634  -1
041635
041635
041635  %=============================================================================
041635  % 29.3        B A T C H  T A B L E
041635  %
041635  BCHTA, 0
041636  "8BCH1; 0;BCH01;1236;1237
041642  "
041642  BNOBA=*
041642          -1
041643
041643
```

```
041643
041643  %=============================================================================
041643  % 29.5        R T - D E S C R I P T I O N  T A B L E
041643  %
041643  % RT-PROGRAMS
041643
041643  NXRTP=0
041643  DUMMY, 0;1000;0;0;0;0;DMSTR;0
041653         BEXQU-5BWLI;0;12;DUMMY
041657         0;0;0
041662
041662  ACRTP

041662         NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
041662  STSIN, 0;1377;0;0;0;0;START;2403 %START PROGRAM
041672         DUMMY;2403;12;STSIN
041676         0;5BFPA@10;0
041701
041701  ACRTP

041701         NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
041701  RTERR, 0;1100;0;0;0;0;SRTER;2414
041711         0;0;100000;RTERR
041715         0;0;0
041720
041720  ACRTP

041720         NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
041720
041720  1SWAP, 0;1277;0;0;0;0;SSSWP;0
041730         0;0;12;1SWAP
041734         0;0;0
041737
041737  ACRTP

041737         NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
041737
041737  "8DILG
041737  RTDIL, 0;1300;0;0;0;0;DILRT;DILGS
041747         0;0;17612;RTDIL
041753         0;0;0
041756  ACRTP

041756         NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
041756
041756  "8BACS
041756  BPTMP, 0;1077;0;0;0;0;9BPTM;0
041766         0;0;17612;BPTMP                  % ON PIT 3
041772         0;0;0
041775  ACRTP

041775         NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
041775  "
041775
041775  RTSLI, 0;1200;0;0;0;0;STSLI;0
042005         0;0;17612;RTSLI                  % ON PIT 3
042011         0;0;0
042014
042014  ACRTP
```

```
042014          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042014
042014  ACCRT,  0;1100;0;0;0;0;117776;32
042024          0;0;12;ACCRT
042030          0;0;0
042033  ACRTP

042033          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042033
042033  TERMP,  0;1200;0;0;0;0;STERM;0
042043          0;0;0;TERMP
042047          0;0;0
042052  ACRTP

042052          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042052  "8N500
042052  5SWAP,  0;1100;0;0;0;0;5SWRT;0          % ND-500 SWAPPER
042062          0;0;0;5SWAP
042066          0;0;0;
042071
042071  ACRTP

042071          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042071
042071  "
042071
042071  RWRT1,  0;1100;0;0;0;0;TRT1;0
042101          0;0;0;RWRT1
042105          0;0;0
042110
042110  ACRTP

042110          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042110  RWRT2,  0;1100;0;0;0;0;TRT2;0
042120          0;0;0;RWRT2
042124          0;0;0
042127
042127  ACRTP

042127          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042127  "8MT1
042127  RWRT3,  0;1100;0;0;0;0;TRT3;0
042137          0;0;0;RWRT3
042143          0;0;0
042146
042146  ACRTP

042146          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042146  "8FDI1+8BFD1
042146  RWRT9,  0;1100;0;0;0;0;TRT9;0
042156          0;0;0;RWRT9
042162          0;0;0
042165  ACRTP

042165          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042165  "8RFAC
042165  RTRFA,  0;1100;0;0;0;0;SRRT1;0
042175          0;0;0;RTRFA
042201          0;0;0
042204  ACRTP
```

```
042204          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042204  % ***********
042204  "SLP1+SLP2+CSP12+SLP3+SLP4+SLP5+SLP6+SLP7+SLP8+SLP9+SLP10+SLP11+SLP12+SLP13+SLP14+SLP15+8COSP+SLP16+SLP17+C
042204  DUMM2, 0;1000;0;0;0;0;1;0
042214          0;0;0;DUMM2
042220          0;0;0
042223  ACRTP

042223          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042223  "SLP1+CSP12
042223  SPRT1, 0;1054;0;0;0;0;SPPR1;SPSGM
042233          0;0;2;SPRT1
042237          0;0;0
042242  ACRTP

042242          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042242  "
042242  "8COSP
042242  COSPO, 0;1054;0;0;0;0;COSDA;0
042252          0;0;40002;COSPO
042256          0;0;0
042261  ACRTP

042261          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042261  "
042261  % ***********
042261
042261
042261  "8DLP1+8DVE1
042261  RWRT11,0;1100;0;0;0;0;TRT11;0
042271          0;0;0;RWRT11
042275          0;0;0
042300  ACRTP

042300          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042300
042300  "IBL01
042300  RWRT13,0;1100;0;0;0;0;TRT13;0
042310          0;0;0;RWRT13
042314          0;0;0
042317  ACRTP

042317          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042317
042317  RWRT20,0;1100;0;0;0;0;TRT20;0
042327          0;0;0;RWRT20
042333          0;0;0
042336  ACRTP

042336          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042336
042336  "BADAD
042336  % BAD ADMINISTRATOR RT-DESCRIPTION
042336  TADAD, 0;1040;0;0;0;0;110000;5BADM
042346          0;0;0;TADAD
042352          0;0;0
042355  ACRTP

042355          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
```

```
042355
042355  "
042355  "8F5UD;                                  % RT-PROG FOR FAST UDMA FROM ND-500
042355  9FPUD=*
042355  "8UD01+8VI01 8F5UD;GUDRT 01
042355  UDR01, 0;1100;0;0;0;0;URT01;0
042365         0;0;017612;UDR01;0;0;0             % RING 2 ,PT=3,ATP=3
042374  )FILL
042374
042374  ACRTP

042374         NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042374  "8F5UD
042374  9LPUD=*
042374
042374  "
```

```
042374
042374  % BACKGROUND PROGRAMS
042374  GSEG=22003
042374  'SLP1
042374  XYS=GSEG; )KILL GSEG; GSEG=XYS+400; )KILL XYS
042374  'BCOSP; XYS=GSEG; )KILL GSEG; GSEG=XYS+400; )KILL XYS
042374   BDILG; XYS=GSEG; )KILL GSEG; GSEG=XYS+400; )KILL XYS
042374
042374   BPR01
042374  XYS=GSEG; )KILL GSEG; GSEG=XYS+1000; )KILL XYS
042374   BN500
042374  XYS=GSEG; )KILL GSEG; GSEG=XYS+400; )KILL XYS
042374
042374  'BRFAC; XYS=GSEG; )KILL GSEG; GSEG=XYS+400; )KILL XYS;
042374
042374
042374   BDB01+BDB02+BDB03+BDB04+BDB05+BDB06+BDB07+BDB08+BDB09
042374  XYS=GSEG; )KILL GSEG; GSEG=XYS+400; )KILL XYS
042374
042374
```

```
042374  %
042374
042374  BBP BAK01
042374
042374  BAK01,  0;1100;0;0;0;0;9ENTO;GSEG
042404          0;0;100000;*-13
042410          0;0;0
042413  XYS=GSEG;)KILL GSEG;GSEG=XYS+1000;)KILL XYS
042413
042413  ACRTP

042413          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042413  "8BP5
042413  BBP BAK05
042413
042413  BAK05,  0;1100;0;0;0;0;9ENTO;GSEG
042423          0;0;100000;*-13
042427          0;0;0
042432  XYS=GSEG;)KILL GSEG;GSEG=XYS+1000;)KILL XYS
042432
042432  ACRTP

042432          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042432  "8BP6
042432  BBP BAK06
042432
042432  BAK06,  0;1100;0;0;0;0;9ENTO;GSEG
042442          0;0;100000;*-13
042446          0;0;0
042451  XYS=GSEG;)KILL GSEG;GSEG=XYS+1000;)KILL XYS
042451
042451  ACRTP

042451          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042451  "
042451  2THSS=*
042451  "8BCH1
042451  BBP BCH01
042451
042451  BCH01,  0;1100;0;0;0;0;9ENTO;GSEG
042461          0;0;100000;*-13
042465          0;0;0
042470  XYS=GSEG;)KILL GSEG;GSEG=XYS+1000;)KILL XYS
042470
042470  ACRTP

042470          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042470  "
042470  9LBPR=*         % LAST BACKGROUND PROGRAM
042470  THISS=*         % TO SEPERATE BC-PROGRAMS FROM RU-PROGRAMS IN TESTS. (NORD-NET)
042470
042470
042470  TIMRT,  0;1200;0;0;0;62;TIMER;0
042500          0;0;0;TIMRT
042504          0;0;0
042507  ACRTP

042507          NYRTP=NXRTP; )KILL NXRTP; NXRTP=NYRTP+1; )KILL NYRTP
042507
```

```
042507
042507
042507
042507
042507
042507  RTBES=*
042507  8RT2=8RTN@4-8RTN                          % (8RTN*20)-8RTN
042507  *+8RT2/
043240
043240   )LINE
043240   %%%%%%%%%%%%  M A C R O E S  -  S I N 4  -  G E N  %%%%%%%%%%%%%%%%%%%%
043240
043240   %%     MACROES USED ON SIN4-GEN
043240   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
043240   %%      PREVIOUSLY USED MACROES ARE KILLED
043240   )KILL BBP JONAC RTUP RWCR RWCW BASFI ACRTP
043240   )KILL VDRT VPRT GUDRT
043240
043240   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
043240
043240   % MACRO FOR SEGMENT TABLE INITIAL SET-UP.
043240
043240   )MCDEF MSGLE $SGLE
        XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
        )KILL SGLEN CSGLE; SGLEN=$SGLE ; CSGLE=SGLEN@10
]
043240
043240   % MACRO FOR SYSTEM SEGMENTS
043240
043240   )MCDEF GSYS
        0;0;SYSSZ+SYSST;GBSEG;162033
GSSEG=GBSEG+5SSSZ;)KILL GBSEG
XY=NNBRT; )KILL NNBRT; NNBRT=XY+1; )KILL XY
]
043240
043240   % MACRO FOR USER SEGMENTS
043240
043240   )MCDEF GUSE
        0;0;LBSEG+FLBPA;GSSEG;160013
GBSEG=GSSEG+LOADR;)KILL GSSEG
]
043240
043240
043240
043240   % FILE SYSTEM MACROS:
043240
043240   "-8S3C
043240   % MACRO FOR DEVICE BUFFER;
043240   )MCDEF FDVBU $DKLDN
F$DKLDN , 0 ; 0 ; *-2 ; 2
-1 ; -1 ; -1 ; $DKLDN ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; *-1 ; *-7 ; *-5 ; *-7 ; 0 ; 0 ; 0
XBU=5BUFA+1;)KILL 5BUFA; 5BUFA=XBU; )KILL XBU
]
043240
043240   "
043240
043240   % MACRO FOR DIRECTORY ENTRY:
043240
043240   )MCDEF DDRTB $DKDVN,$DKNAN,$DILDN,$DKRES
0; $DKDVN ; $DKNAN ; $DILDN ; $DKRES ;
```

```
0;0;0;0;0;0;0;0;                                  % DIRECTORY NAME
0;0;0;0;0;0;0;0
0;0
]
043240
043240  )MCDEF MDRTB $DKDVN,$DKNAN,,$DILDN,$DKRES
20000; $DKDVN ; $DKNAN ; $DILDN ; $DKRES ; 0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
]
043240
043240  % MACRO FOR BIT FILE BUFFER:
043240  )MCDEF FBFBU $BFLDN
BINDX ; -1; $BFLDN ; *+20/
XBINX=BINDX+1;)KILL BINDX;BINDX=XBINX;)KILL XBINX
]
043240
043240
043240
043240  %       MACRO FOR SPOOLING DATAFIELD IN RESIDENT
043240  %
043240  %       $SRT        SPOOLING RT PROGRAM NAME.
043240  %       $SLNR       LOGICAL DEVICE NUMBER OF DEVICE TO WHICH DATA IS SPOOLED.
043240  %       $QSEG       SEGMENT NUMBER WHERE QUEUE IS HELD (= QUEUE NUMBER).
043240  %       $INXX       SPOOLING INDEX
043240  %       $QSEM       QUEUE SEMAPHORE - USED TO LOCK QUEUE.
043240  %       $QIOS       QUEUE I/O SEMAPHORE - USED TO WAIT FOR SOMETHING TO BE INSERTED IN QUEUE.
043240
043240  )MCDEF SPOOD $SPPR,$SRT,$SLNR,$QSEG,$INXX,$QSEM,$QIOS

$SPPR ,JPL I *1; SPORT; $SRT ;$SLNR ;$QSEG ;$QSEM ;$QIOS ;0;0;$INXX ;

]
043240
043240  %       MACRO FOR SPOOLING QUEUE SEGMENT
043240  )MCDEF SPOOS
9GBSG=SPSGL@10
0;0;9GBSG+60;GBSEG;162003                              %       SPOOLING QUEUE SEGMENT
GSSEG=GBSEG+SPSGL; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
)KILL 9GBSG

]
043240
043240
043240  )MCDEF SPBUF $LABE
0;$LABE ;0;0;162005
]
043240
043240  % ********** %
043240  % MACRO FOR REMOTE FILE TRANSFER SEGMENT
043240  )MCDEF RFTS
        0;0;1074;GBSEG;162003
        GSSEG=GBSEG+2;)KILL GBSEG;GBSEG=GSSEG;)KILL GSSEG
]
043240
043240  % MACRO FOR DEBUGGER DATA SEGMENT
043240  )MCDEF GDEBU
        0;0;6164;GBSEG;140043
        GSSEG=GBSEG+14;)KILL GBSEG;GBSEG=GSSEG;)KILL GSSEG
]
043240  "8N500
043240  %*****************************************************************
```

```
043240  %       M A C R O E S   F O R   N D 5 0 0   S Y S T E M
043240  %
043240  % MACRO TO ALLOCATE A ND500 PROCESS DESCRIPTION
043240
043240   )MCDEF PRSDC
        0;0;*-2;2;4000+N5DSG;0;RTBAK;0
    *   0;0;0;0;0;0;70;N5DSG+1
        0;0;0;0;0;0;0;0
        0;0;0;0;0;0;0;70
        70;20;0;0;0;0
X5=NN5MS; )KILL NN5MS; NN5MS=X5+5MESS; )KILL X5
X5=N5DSG; )KILL N5DSG; N5DSG=X5+2; )KILL X5
X5=ACCE5+15; )KILL ACCE5; ACCE5=X5; )KILL X5
X5=NU5PR+1; )KILL NU5PR; NU5PR=X5; )KILL X5
)
043240
043240  % MACRO FOR ALLOCATING TWO SEGMENTS FOR EACH ND500 PROCESS
043240
043240   )MCDEF M5SGS
0;0;2444;GBSEG;162043
0;0;0;0;40
GSSEG=GBSEG+5; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
TEMP=SSSEG; )KILL SSSEG; SSSEG=TEMP+2; )KILL TEMP
)
043240
043240   8RFAC
043240  % ***  MACRO TO ALLOCATE DATA SEGMENTS FOR REMOTE FILE ACCESS ***
043240   )MCDEF RFDSG
       0;0;2072;GBSEG;162003
GSSEG=GBSEG+4; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
TEMP=SSSEG; )KILL SSSEG; SSSEG=TEMP+1; )KILL TEMP
)
043240
```

```
043240  )LINE
043240  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  S I N 4  -  G E N  %%%%%%%%%%%%%%%%%%%%%%%%%%
043240
043240  %=============================================================================
043240  % 29.6        S E G M E N T  T A B L E
043240  %
043240  % CALCULATE VALUES FOR ALLOCATING (LOGICAL AND PHYSICAL ADDRESSES) SEGMENTS
043240
043240  % LOGICAL START OF SYSTEM SEGMENT
043240  SYSST=BGSYS@-12
043240  % SIZE OF SYSTEM SEGMENT (5 PAGES - IN LEFT BYTE)
043240  SYSSZ=5SSSZ@10
043240  5NBSG=2        % NUMBER OF SEGMENTS FOR EACH BACKGROUND PROCESS
043240  5SDSP=1002     % SEGM. DISPLACEMENT (2 FOR SYSTEM AND USER SEGM.)
043240  "8RFAC
043240  5FDSZ=4        % SIZE OF FILE USER DATA SEGMENT
043240  FRSSZ=26       % SIZE OF FILE USER REENTRANT SEGMENT
043240  FDSTR=72       % START PAGE OF FILE USER DATA SEGMENT
043240  "
043240  5ESSZ=2        % SIZE OF ERROR SEGMENT
043240  SGLEN=0        % SEGMENT LENGTH IN PAGES
043240  SGMAD=0        % MASS STORAGE ADDRESS IN PAGES OF THE SEGMENT
043240  NNBRT=0        % NUMBER OF BACKGROUND PROGRAMS
043240
043240  %=============================================================================
043240  %       S E G M E N T  T A B L E
043240
043240  XSEGS=*
043240          0;0;0;0;0                       % RESIDENT
043245  XSGRT, -1
043246  BPAG1, XCORM;177000;0;161000            % DUMMY SEGMENT
043252
043252          MSGLE 77
043252
043252          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043252          )KILL SGLEN CSGLE; SGLEN=77; CSGLE=SGLEN@10
043252
043252  SG2,    0;0;CSGLE+100;SGMAD;162003      % MEMORY IMAGE
043257          SGLEN:000077      SGMAD:000000
043257          SG02S=00@12
043257
043257          MSGLE 32
043257
043257          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043257          )KILL SGLEN CSGLE; SGLEN=32; CSGLE=SGLEN@10
043257
043257  SG3,    0;0;CSGLE+044;SGMAD;162003      % COMMAND SEGMENT
043264          SGLEN:000032      SGMAD:000077
043264          SG03S=44@12
043264
043264          MSGLE 20
043264
043264          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043264          )KILL SGLEN CSGLE; SGLEN=20; CSGLE=SGLEN@10
043264
043264  SG4,    0;0;CSGLE+044;SGMAD;62003       % REENTRAN RT-LOADER SEGMENT
043271          SGLEN:000020      SGMAD:000131
043271          SG04S=44@12
043271
043271          MSGLE 5ESSZ
```

```
043271
043271          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043271          )KILL SGLEN CSGLE; SGLEN=5ESSZ; CSGLE=SGLEN@10
043271
043271  SG5,    0;0;CSGLE+SYSST;SGMAD;162003     % ERROR PROGRAM SYSTEM SEGMENT (2 PAGES)
043276          SGLEN:000002      SGMAD:000151
043276          SG05S=SYSST@12
043276
043276          MSGLE 14
043276
043276          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043276          )KILL SGLEN CSGLE; SGLEN=14; CSGLE=SGLEN@10
043276
043276  SG6,    0;0;CSGLE+044;SGMAD;162003       % FILE-SYSTEM COMMON SEGMENT
043303          SGLEN:000014      SGMAD:000153
043303          SG06S=44@12
043303
043303          MSGLE 22
043303
043303          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043303          )KILL SGLEN CSGLE; SGLEN=22; CSGLE=SGLEN@10
043303
043303  SG7,    0;0;CSGLE+044;SGMAD;162003       % DMAC SEGMENT
043310          SGLEN:000022      SGMAD:000167
043310          SG07S=44@12
043310
043310          MSGLE 100
043310
043310          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043310          )KILL SGLEN CSGLE; SGLEN=100; CSGLE=SGLEN@10
043310
043310  SG10,   0;0;CSGLE+200;SGMAD;162003       % RTFIL SEGMENT
043315          SGLEN:000100      SGMAD:000211
043315          SG10S=00@12
043315
043315          MSGLE 10
043315
043315          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043315          )KILL SGLEN CSGLE; SGLEN=10; CSGLE=SGLEN@10
043315
043315  SG11,   0;0;CSGLE+100;SGMAD;162003       % ERROR LOG SEGMENT
043322          SGLEN:000010      SGMAD:000311
043322          SG11S=00@12
043322
043322          MSGLE 16
043322
043322          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043322          )KILL SGLEN CSGLE; SGLEN=16; CSGLE=SGLEN@10
043322
043322  SG12,   0;0;CSGLE+060;SGMAD;162003       % INITIAL REENTRANT FILE SYS. SEGMENT #2
043327          SGLEN:000016      SGMAD:000321
043327          SG12S=60@12
043327
043327          MSGLE  35
043327
043327          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043327          )KILL SGLEN CSGLE; SGLEN= 35; CSGLE=SGLEN@10
043327
043327  SG13,   0;0;CSGLE+127;SGMAD;162003       % INITIAL RT-LOADER SEGMENT
043334          SGLEN:000035      SGMAD:000337
```

```
043334          SG13S=27@12
043334
043334          MSGLE 6
043334
043334          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043334          )KILL SGLEN CSGLE; SGLEN=6; CSGLE=SGLEN@10
043334
043334  SG14,   0;0;CSGLE+044;SGMAD;162003      % ERROR PROGRAM SEGMENT
043341          SGLEN:000006      SGMAD:000374
043341          SG14S=44@12
043341
043341          MSGLE 32
043341
043341          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043341          )KILL SGLEN CSGLE; SGLEN=32; CSGLE=SGLEN@10
043341
043341  SG15,   0;0;CSGLE+044;SGMAD;162003      % INITIAL SINTRAN SERVICE AND MAIL
043346          SGLEN:000032      SGMAD:000402
043346          SG15S=44@12
043346
043346          MSGLE 10
043346
043346          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043346          )KILL SGLEN CSGLE; SGLEN=10; CSGLE=SGLEN@10
043346
043346  SG16,   0;0;CSGLE+064;SGMAD;162003      % INITIAL NORD-NET SEGMENT
043353          SGLEN:000010      SGMAD:000434
043353          SG16S=64@12
043353
043353          MSGLE 14
043353
043353          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043353          )KILL SGLEN CSGLE; SGLEN=14; CSGLE=SGLEN@10
043353
043353  SG17,   0;0;CSGLE+100;SGMAD;162000      % INITIAL PIT3-SEGMENT
043360          SGLEN:000014      SGMAD:000444
043360          SG17S=00@12
043360
043360          MSGLE 14                % RESERVE MASS STORAGE SPACE FOR INITIAL SPOOLING SEGMENT
043360
043360          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043360          )KILL SGLEN CSGLE; SGLEN=14                ; CSGLE=SGLEN@10
043360
043360          MSPIA=SGMAD
043360
043360          MSGLE 14                % RESERVE MASS STORAGE SPACE FOR SPOOLING SEGMENT
043360
043360          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043360          )KILL SGLEN CSGLE; SGLEN=14                ; CSGLE=SGLEN@10
043360
043360          MSPOA=SGMAD
043360
043360  'BN500;MSGLE 100
043360
043360          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043360          )KILL SGLEN CSGLE; SGLEN=100; CSGLE=SGLEN@10
043360
043360  ..
043360  SG20,   0;0;CSGLE+200;SGMAD;162003      % ND-500 STANDARD DOMAIN SEGMENT
043365          SGLEN:000100      SGMAD:000510
```

```
043365          SG20S=00@12
043365
043365    "8N500;MSGLE 100
043365
043365          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043365          )KILL SGLEN CSGLE; SGLEN=100 ; CSGLE=SGLEN@10
043365
043365    "
043365    SG21,   0;0;CSGLE+200;SGMAD;162003       % ND-500 NAME TABLES SEGMENT
043372          SGLEN:000100      SGMAD:000610
043372          SG21S=00@12
043372
043372    "8RFAC;MSGLE 26;
043372
043372          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043372          )KILL SGLEN CSGLE; SGLEN=26; CSGLE=SGLEN@10
043372     SG22S=44@12;"
043372
043372    SG22,   0;0;CSGLE+044;SGMAD;62003        % REENTRANT FILE USER SEGMENT #1
043377          SGLEN:000026      SGMAD:000710
043377
043377          MSGLE 32
043377
043377          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043377          )KILL SGLEN CSGLE; SGLEN=32; CSGLE=SGLEN@10
043377
043377    SG23,   0;0;CSGLE+044;SGMAD;162003       % SERVICE-PROGRAM & MAIL
043404          SGLEN:000032      SGMAD:000736
043404          SG23S=44@12
043404
043404          MSGLE 16
043404
043404          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043404          )KILL SGLEN CSGLE; SGLEN=16; CSGLE=SGLEN@10
043404
043404    SG24,   0;0;CSGLE+060;SGMAD;062003       % FILE SYSTEM REENTRANT SEGMENT #1
043411          SGLEN:000016      SGMAD:000770
043411          SG24S=60@12
043411
043411          MSGLE 16
043411
043411          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043411          )KILL SGLEN CSGLE; SGLEN=16; CSGLE=SGLEN@10
043411
043411    SG25,   0;0;CSGLE+060;SGMAD;062003       % FILE SYSTEM REENTRAN SEGMENT 2
043416          SGLEN:000016      SGMAD:001006
043416          SG25S=60@12
043416
043416    "8RFAC;MSGLE 26;
043416
043416          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043416          )KILL SGLEN CSGLE; SGLEN=26; CSGLE=SGLEN@10
043416     SG26S=44@12;"
043416
043416    SG26,   0;0;CSGLE+044;SGMAD;62003        % REENTRANT FILE USER SEGMENT #2
043423          SGLEN:000026      SGMAD:001024
043423
043423
043423    "8NNET;MSGLE 0;
043423
```

```
043423          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043423          )KILL SGLEN CSGLE; SGLEN=0; CSGLE=SGLEN@10
043423   SG27S=00@12;"
043423  SG27,   0;0;CSGLE+064;SGMAD;162000      % NORD-NET
043430          SGLEN:000000      SGMAD:001052
043430
043430  "8N500;MSGLE 25;
043430
043430          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043430          )KILL SGLEN CSGLE; SGLEN=25; CSGLE=SGLEN@10
043430   SG30S=51@12;"
043430
043430  SG30,   0;0;CSGLE+051;SGMAD;062003      % ND-500 REENTRANT SYSTEM MONITOR #1
043435          SGLEN:000025      SGMAD:001052
043435
043435  "8N500;MSGLE 25;
043435
043435          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043435          )KILL SGLEN CSGLE; SGLEN=25; CSGLE=SGLEN@10
043435   SG31S=51@12;"
043435
043435  SG31,   0;0;CSGLE+051;SGMAD;062003      % ND-500 REENTRANT SYSTEM MONITOR #2
043442          SGLEN:000025      SGMAD:001077
043442
043442          MSGLE 10
043442
043442          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043442          )KILL SGLEN CSGLE; SGLEN=10; CSGLE=SGLEN@10
043442
043442  SG32,   0;0;CSGLE+044;SGMAD;162003      % RT-ACCOUNTING
043447          SGLEN:000010      SGMAD:001124
043447          SG32S=44@12
043447
043447          MSGLE 0
043447
043447          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043447          )KILL SGLEN CSGLE; SGLEN=0; CSGLE=SGLEN@10
043447
043447  SG33,   0;0;CSGLE+000;SGMAD;162003      % X-MESSAGE "POF"
043454          SGLEN:000000      SGMAD:001134
043454
043454          MSGLE 0
043454
043454          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043454          )KILL SGLEN CSGLE; SGLEN=0; CSGLE=SGLEN@10
043454
043454  SG34,   0;0;CSGLE+000;SGMAD;162003      % X-MESSAGE PROGRAM
043461          SGLEN:000000      SGMAD:001134
043461
043461          MSGLE 0
043461
043461          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043461          )KILL SGLEN CSGLE; SGLEN=0; CSGLE=SGLEN@10
043461
043461  SG35,   0;0;CSGLE+000;SGMAD;162003      % X-MESSAGE
043466          SGLEN:000000      SGMAD:001134
043466
043466  "BADAD;MSGLE 12;
043466
043466          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
```

```
043466          )KILL SGLEN CSGLE; SGLEN=12; CSGLE=SGLEN@10
043466    SG36S=44@12;"
043466
043466  SG36,   0;0;CSGLE+044;SGMAD;162003       % TADADM SEGMENT
043473          SGLEN:000012      SGMAD:001134
043473
043473          MSGLE 100
043473
043473          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043473          )KILL SGLEN CSGLE; SGLEN=100; CSGLE=SGLEN@10
043473
043473  SG37,   0;0;CSGLE+100;SGMAD;162003       % RT-LOADER DATA SEGMENT
043500          SGLEN:000100      SGMAD:001146
043500          SG37S=00@12
043500
043500  "8RFAC;MSGLE 4;
043500
043500          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043500          )KILL SGLEN CSGLE; SGLEN=4; CSGLE=SGLEN@10
043500    SG40S=72@12;"
043500
043500  SG40,   0;0;CSGLE+072;SGMAD;162003       % FILE USER DATA SEGMENT FOR RT.
043505          SGLEN:000004      SGMAD:001246
043505
043505          MSGLE 14
043505
043505          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043505          )KILL SGLEN CSGLE; SGLEN=14; CSGLE=SGLEN@10
043505
043505  SG41,   0;0;CSGLE+S41FP;SGMAD;162000     % PIT3-SEGMENT
043512          SGLEN:000014      SGMAD:001252
043512          % S41FP IS DEFINED ON THE "PIT3-SEGMENT"
043512  S41FL=*-1       % ADDR OF SG41.FLAG
043512
043512          MSGLE 0
043512
043512          XY=SGMAD; )KILL SGMAD; SGMAD=XY+SGLEN; )KILL XY
043512          )KILL SGLEN CSGLE; SGLEN=0; CSGLE=SGLEN@10
043512
043512
043512          SG42S=44@12
043512  SG42,   0;0;6044;MSPIA;162003           % SAVE SEGMENT FOR SPOOLING PROGRAM
043517          MSPIA;000460
043517
043517          SG43S=44@12
043517  SG43,   0;0;6044;MSPOA;162003           % SPOOLING PROGRAM SEGMENT
043524          MSPOA:000474
043524
043524  )KILL MSPIA MSPOA
043524
043524          SGMAD:001266
043524
043524  SSSEG=44      % FIRST CONFIGURATION DEPENDANT SEGMENT
043524  GBSEG=SGMAD   % NEXT FREE MASS STORAGE ADDRESS ON SEGFILO
043524
043524  "8DILG
043524  DILGS=SSSEG;)KILL SSSEG; SSSEG=DILGS+1
043524  0;0;444;GBSEG;162000
043531  GSSEG=GBSEG+1;)KILL GBSEG;GBSEG=GSSEG;)KILL GSSEG
043531   SLP1
```

```
043531  SPSG1=SSSEG;)KILL SSSEG;SSSEG=SPSG1+1
043531  SPOOS

043531  9GBSG=SPSGL@10
043531  0;0;9GBSG+60;GBSEG;162003                                    %      SPOOLING QUEUE SEGMENT
043536  GSSEG=GBSEG+SPSGL; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
043536  )KILL 9GBSG
043536
043536  "8COSP
043536  COSEG=SSSEG;)KILL SSSEG;SSSEG=COSEG+1
043536  SPOOS

043536  9GBSG=SPSGL@10
043536  0;0;9GBSG+60;GBSEG;162003                                    %      SPOOLING QUEUE SEGMENT
043543  GSSEG=GBSEG+SPSGL; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
043543  )KILL 9GBSG
043543
043543  "
043543
043543  "8PRO1
043543  % SEGMENTS FOR ND500 MONITOR
043543  %
043543  % 2 SEGMENTS FOR EACH PROCESS, DATA SEGMENT AND FILE-TRANSFER SEGMENT
043543  %
043543
043543  N5DSG= SSSEG-2
043543  F5DSG= SSSEG
043543  M5SGS                                                                    --

043543  0;0;2444;GBSEG;162043
043550  0;0;0;0;40
043555  GSSEG=GBSEG+5; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
043555  TEMP=SSSEG; )KILL SSSEG; SSSEG=TEMP+2; )KILL TEMP
043555  "8N500
043555  L5DSG=SSSEG
043555  0;0;2444;GBSEG;162043
043562  GSSEG=GBSEG+5; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
043562  TEMP=SSSEG; )KILL SSSEG; SSSEG=TEMP+1; )KILL TEMP
043562  "
043562  %  ***************  DATA SEGMENTS FOR REMOTE FILE ACCESS;    ******************
043562  "8RFAC
043562  DSSNM=SSSEG      % SEGMENT NUMBER OF FIRST DATA SEGMENT FOR REMOTE FILE ACCESS
043562  "8RFAC; RFDSG;
043562          0;0;2072;GBSEG;162003
043567  GSSEG=GBSEG+4; )KILL GBSEG; GBSEG=GSSEG; )KILL GSSEG
043567  TEMP=SSSEG; )KILL SSSEG; SSSEG=TEMP+1; )KILL TEMP
043567
043567  "
043567
043567  SGNUM=SSSEG                      % FIRST SEGMENT NO. FOR SYMBOLIC DEBUGGER
043567  "8DB01+8DB02+8DB03+8DB04+8DB05+8DB06+8DB07+8DB08+8DB09
043567  GDEBU

043567          0;0;6164;GBSEG;140043
043574          GSSEG=GBSEG+14;)KILL GBSEG;GBSEG=GSSEG;)KILL GSSEG
043574  "
043574
043574
043574  % SYSTEM SEGMENT 1:
043574  GSYS
```

```
043574
043574          0;0;SYSSZ+SYSST;GBSEG;162033
043601  GSSEG=GBSEG+5SSSZ;)KILL GBSEG
043601  XY=NNBRT; )KILL NNBRT; NNBRT=XY+1; )KILL XY
043601  % BACKGROUND SEGMENT 1:
043601  GUSE

043601          0;0;LBSEG+FLBPA;GSSEG;160013
043606  GBSEG=GSSEG+LOADR;)KILL GSSEG
043606
043606  "8BP5
043606  GSYS

043606
043606          0;0;SYSSZ+SYSST;GBSEG;162033
043613  GSSEG=GBSEG+5SSSZ;)KILL GBSEG
043613  XY=NNBRT; )KILL NNBRT; NNBRT=XY+1; )KILL XY
043613  GUSE

043613          0;0;LBSEG+FLBPA;GSSEG;160013
043620  GBSEG=GSSEG+LOADR;)KILL GSSEG
043620  "8BP6
043620  GSYS

043620
043620          0;0;SYSSZ+SYSST;GBSEG;162033
043625  GSSEG=GBSEG+5SSSZ;)KILL GBSEG
043625  XY=NNBRT; )KILL NNBRT; NNBRT=XY+1; )KILL XY
043625  GUSE

043625          0;0;LBSEG+FLBPA;GSSEG;160013
043632  GBSEG=GSSEG+LOADR;)KILL GSSEG
043632  "8BCH1
043632  GSYS

043632
043632          0;0;SYSSZ+SYSST;GBSEG;162033
043637  GSSEG=GBSEG+5SSSZ;)KILL GBSEG
043637  XY=NNBRT; )KILL NNBRT; NNBRT=XY+1; )KILL XY
043637  GUSE

043637          0;0;LBSEG+FLBPA;GSSEG;160013
043644  GBSEG=GSSEG+LOADR;)KILL GSSEG
043644  "
043644  SEGTX=*
043644  *8SGN 8SGN 8SGN 8SGN 8SGN/
044135  9ESGT=*
044135
044135
```

```
044135
044135  %=============================================================================
044135  %              T I M E   S L I C E   T A B L E S
044135  %
044135  "BBACS
044135  XNOBP=NNBRT
044135  "
044135  % TIMESLICE STATUS TABLE
044135  TSLST=*
044135  TSLST+XNOBP/
044141  TSLST<*; )ZERO
044141
044141  % TIMSLICE TIME-COUNTER TABLE
044141  TSLCO=*
044141  TSLCO+XNOBP/
044145  MXM1=-1
044145  TSLCO<*; )ZERO MXM1; )KILL MXM1
044145
044145  % TIMESLICE "NUMBER OF TIMESLICE UNITS BEFORE NEXT TIMESLICE ELEMENT" TABLE
044145  TSLNT=*
044145  TSLNT+XNOBP/
044151  TSLNT<*; )ZERO
044151
044151
044151
```

```
044151
044151  "8N500
044151  %=============================================================================
044151  %       N D 5 0 0   P R O C E S S   D E S C R I P T I O N S
044151  %
044151
044151  %       N D 5 0 0   P R O C E S S   D E S C R I P T I O N S
044151
044151  NN5MS= 0
044151  ACCE5= 0
044151  NU5PR= 0
044151
044151  S500S=*; PRSDC

044151          0;0;*-2;2;4000+N5DSG;0;RTBAK;0
044161          0;0;0;0;0;0;70;N5DSG+1
044171          0;0;0;0;0;0;0;0
044201          0;0;0;0;0;0;0;70
044211          70;20;0;0;0;0
044217  X5=NN5MS; )KILL NN5MS; NN5MS=X5+5MESS; )KILL X5
044217  X5=N5DSG; )KILL N5DSG; N5DSG=X5+2; )KILL X5
044217  X5=ACCE5+15; )KILL ACCE5; ACCE5=X5; )KILL X5
044217  X5=NU5PR+1; )KILL NU5PR; NU5PR=X5; )KILL X5
044217  "8PR01; PRSDC

044217          0;0;*-2;2;4000+N5DSG;0;RTBAK;0
044227          0;0;0;0;0;0;70;N5DSG+1
044237          0;0;0;0;0;0;0;0
044247          0;0;0;0;0;0;0;70
044257          70;20;0;0;0;0
044265  X5=NN5MS; )KILL NN5MS; NN5MS=X5+5MESS; )KILL X5
044265  X5=N5DSG; )KILL N5DSG; N5DSG=X5+2; )KILL X5
044265  X5=ACCE5+15; )KILL ACCE5; ACCE5=X5; )KILL X5
044265  X5=NU5PR+1; )KILL NU5PR; NU5PR=X5; )KILL X5
044265  "8N500
044265  S500E=*;PRSDC

044265          0;0;*-2;2;4000+N5DSG;0;RTBAK;0
044275          0;0;0;0;0;0;70;N5DSG+1
044305          0;0;0;0;0;0;0;0
044315          0;0;0;0;0;0;0;70
044325          70;20;0;0;0;0
044333  X5=NN5MS; )KILL NN5MS; NN5MS=X5+5MESS; )KILL X5
044333  X5=N5DSG; )KILL N5DSG; N5DSG=X5+2; )KILL X5
044333  X5=ACCE5+15; )KILL ACCE5; ACCE5=X5; )KILL X5
044333  X5=NU5PR+1; )KILL NU5PR; NU5PR=X5; )KILL X5
044333  "
044333  %=============================================================================
044333  % 29.7        R T C O M M O N   T A B L E
044333  %
044333  CCNOX,CCNO
044334  CCSTA=*;0
044335  CCTAB=*
044335  ++(CNO+CCNO+20/
044357  -1
044360
044360
044360  %=============================================================================
044360  % 29.8        I O X   T A B L E
044360  %
```

```
044360  % IOX-ADDRESS CHECK TABLE
044360  IOXTA=*
044360  *+8IOXT/
044370  EIOXT, -1                       % END OF IOX-TABLE
044371
044371
044371  %=============================================================================
044371  % 29.9        F I L E S Y S T E M   D E F I N I T I O N S
044371  %
044371
044371
044371  % OPEN FILE NUMBER TABLE AND BUFFER POOL DEFINITIONS:
044371
044371  FMAX 5NPOO                      % TO UNDEFINE FMAX AND 5NPOO IF NOT DEFINED
044372  "FMAX
044372  FMAX=34                         % DEFAULT NUMBER OF OPEN FILES IS 28
044372  "
044372  ENDOP=OPTAB+FMAX+FMAX           % FIRST LOCATION AFTER OPEN FILE TABLE
044372  BPOOL=ENDOP                     % IS ALSO START OF BUFFER POOL FOR OPEN FILES
044372  "5NPOO
044372  5NPOO=34                        % DEFAULT NUMBER OF BUFFERS IS 28
044372                                  % ENOUGH FOR 28 RANDOM OPENED FILES
044372  "
044372  *-1/
044371  )KILL CONTX; CONTX=5NPOO@6+5NPOO+BPOOL     % CONTX=101*5NPOO+BPOOL
044371
044371
044371  S000,  0
044372  SDUMM, DUMMY
044373  S504=504
044373  S505=505
044373  S506=506
044373  S507=507
044373  S511=511
044373  S512=512
044373  S513=513
044373  S514=514
044373  S521=521
044373  S522=522
044373  S523=523
044373  S524=524
044373  S525=525
044373  S554=554
044373  S555=555
044373  S556=556
044373  S557=557
044373  S40D,  (40
044374          S000;SDUMM
044376  S561=561
044376  S41D,  (41
044377          S000;SDUMM
044401  S25D,  (25
044402          S000;SDUMM
044404  S570=570
044404  S571=571
044404  S572=572
044404  S573=573
044404  MOLDS=516
044404  S516=MOLDS
044404  )FILL
```

```
044407
044407   S1101=1101
044407   S1102=1102
044407   S1105=1105
044407   S1106=1106
044407   S1107=1107
044407   S1110=1110
044407   S1112=1112
044407   S1114=1114
044407   S1117=1117
044407   S1120=1120
044407   S1121=1121
044407   S1122=1122
044407   S1123=1123
044407   S1124=1124
044407   S1127=1127
044407   S1130=1130
044407   S1131=1131
044407   S1132=1132
044407   S1133=1133
044407   S1134=1134
044407   S1135=1135
044407   S1142=1142
044407   S1147=1147
044407   S1150=1150
044407   S1151=1151
044407   S1152=1152
044407   S1153=1153
044407   S1154=1154
044407   S1155=1155
044407   S1160=1160
044407   S1161=1161
044407   S1162=1162
044407   S1163=1163
044407   S1164=1164
044407   S1165=1165
044407   S1166=1166
044407   S1225=1225                    %
044407   S1226=1226                    %
044407   S1227=1227                    %
044407   S1230=1230                    % SATELLITE DIRECTORY AND BIT FILE BUFFER LOCKS
044407   S1232=1232                    %  (I.E. FINCH DISK)
044407   S1233=1233                    %
044407   S1234=1234                    %
044407   S1235=1235                    %
044407
044407   S32D,   (32
044410           S000;SDUMM
044412   SX7D,   (1113
044413           S000;SDUMM
044415   S34D,   (34
044416           S000;SDUMM
044420   SX8D,   (1115
044421           S000;SDUMM
044423   )FILL
044427   S563D,  (563
044430           S000;SDUMM
044432   S564D,  (564
044433           S000;SDUMM
044435
```

```
044435  S33D,   (33
044436          S000;SDUMM
044440  )FILL
044443
044443  SX1D,   (1000
044444          S000;SDUMM
044446  SX2D,   (1001
044447          S000;SDUMM
044451  SX3D,   (1002
044452          S000;SDUMM
044454  SX4D,   (1003
044455          S000;SDUMM
044457  SX5D,   (1004
044460          S000;SDUMM
044462  SX6D,   (1005
044463          S000;SDUMM
044465  )FILL
044473  D1232,  (1232
044474          S000;SDUMM
044476  D1233,  (1233
044477          S000;SDUMM
044501  D1234,  (1234
044502          S000;SDUMM
044504  D1235,  (1235
044505          S000;SDUMM
044507  )FILL
044513  D1225,  (1225
044514          S000;SDUMM
044516  D1226,  (1226
044517          S000;SDUMM
044521  D1227,  (1227
044522          S000;SDUMM
044524  D1230,  (1230
044525          S000;SDUMM
044527  )FILL
044533
044533  S1171=1171
044533  S1172=1172
044533  S1177=1177
044533  S1300=1300
044533  S1301=1301
044533  S1333=1333
044533  S1334=1334
044533  S1335=1335
044533  S1336=1336
044533  S1337=1337
044533  S1340=1340
044533  S1341=1341
044533  S1342=1342
044533  % *** VIRTUAL DISK DRIVER LDNs
044533  VDD01=1716
044533  VDD02=1717
044533  VDD03=1720
044533  VDD04=1721
044533  % ***
044533  %       SYMBOL DEFINITIONS FOR MICROPOLIS 5 1/4 INCH WINCHESTER DISK.
044533  S1732=1732
044533  S1733=1733
044533  S1734=1734
044533  S1735=1735
```

```
044533  S1737=1737
044533  S1740=1740
044533  S1741=1741
044533  S1742=1742
044533  S1743=1743
044533  S1744=1744
044533  S1745=1745
044533  S1746=1746
044533  S1747=1747
044533  S1750=1750
044533  S1751=1751
044533  S1752=1752
044533  %
044533  SLDN,   S1142
044534  WSLDN,  1142
044535  WULDN,  1143
044536  WFLDN,  1144
044537  FDFSE,  1177
044540  )FILL
044540
044540  GLDN,   S504            % GENERAL LOCK FOR FILE SYSTEM
044541  MOLDN,  516             % OPEN - CLOSE MONITOR CALL PROGRAM LOCK
044542  ULDN=   S505            % USER FILE BUFFER LOCK
044542  OLDN=   S506            % OBJECT FILE BUFFER LOCK
044542  OFLDN=  S507            % RT-OPEN-FILE-TABLE LOCK
044542  BLDN,   S521            % GET DEVICE BUFFER LOCK
044543
044543
044543
```

```
044543
044543  %=============================================================================
044543  % 29.10      D E V I C E  B U F F E R S
044543  %
044543
044543  % SPECIAL DEVICE BUFFER FOR DIRECT TRANSFER:
044543  5BUFA=0
044543  DTHEA=*
044543  FDVBU 0
044543  F0, 0 ; 0 ; *-2 ; 2
044547  -1 ; -1 ; -1 ; 0; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; *-1 ; *-7 ; *-5 ; *-7 ; 0 ; 0 ; 0
044570  XBU=5BUFA+1;)KILL 5BUFA; 5BUFA=XBU; )KILL XBU
044570
044570
044570  DEVBU=*
044570
044570  )KILL 5BUFA
044570  5BUFA=0
044570  "8BU0+8BU1+8BU2+8BU3+8BU4+8BU5+8BU6+8BU7
044570  FDVBU 1600
044570  F1600, 0 ; 0 ; *-2 ; 2
044574  -1 ; -1 ; -1 ; 1600; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; *-1 ; *-7 ; *-5 ; *-7 ; 0 ; 0 ; 0
044615  XBU=5BUFA+1;)KILL 5BUFA; 5BUFA=XBU; )KILL XBU
044615
044615  "
044615  ENDBU=*
044615
044615
044615  %=============================================================================
044615  % 29.11      S P O O L I N G  D A T A F I E L D S
044615  %
044615  % SPOOLING DATAFIELDS
044615
044615  ENEN,  1
044616  SPTAB=*+1
044616  "SLP1
044616  SPOOD SPPR1,SPRT1,SLD1,SPSG1,1,1136,1137
044616
044616  SPPR1,JPL I *1; SPORT; SPRT1;SLD1;SPSG1;1136;1137;0;0;1;
044630
044630
044630  "8COSP
044630  COSDA, JPL I *1;110000;COSPO;1731;COSEG;2166;2167;0;0;37
044642
044642  "
044642  ENDSP=*+1
044642
044642
044642  CURR=*
044642
044642  110000/110000
110001  ESSTA-1
110002  MCCLD
110003
```

```
110003
110003  %==========================================================================
110003  % 29.13       N A M E   T A B L E
110003  %
110003  NAMTA=*
110003  GLOB=0
110003
110003  "7B1U0+7B1U1+7B1U2+7B1U3
110003  'DISC-75MB-1';0;0;  0;110121;1000;  2001;BABST;515
110021  D75M1=GLOB;)KILL GLOB;GLOB=D75M1+1
110021  "8FDI1+8BFD1
110021  'FLOPPY-DISC-1';0;    0;  232; 400;  20401;FDABS;1146
110037  FLOP1=GLOB;)KILL GLOB;GLOB=FLOP1+1
110037  "8MT1
110037  'MAG-TAPE-1';0;0;     0;     0;     1; 24001;MABST;526
110055  MATA1=GLOB;)KILL GLOB; GLOB=MATA1+1
110055
110055  "
110055  ENDNT=*
110055  FFTAB=*
110055  010400 % FLOPPY FORMAT 0 : SECTOR SIZE 400, DEVICE SIZE 232 PAGES
110056  "8BFD1+8BFD2
110056  022017 % FLOPPY FORMAT 17: SECTOR SIZE 400*2, DEVICE SIZE 232*4 PAGES
110057  "
110057  -1
110060
110060  %==========================================================================
110060  % 29.14       D I R E C T O R Y   T A B L E
110060  %
110060  DIRTA=*                                    % SYMBOL FOR START OF DIRECTORY TABLE.
110060
110060  "7B1U0
110060  DDRTB 1100,D75M1,S1101,0
110060  0; 1100; D75M1; S1101; 0;
110065  0;0;0;0;0;0;0;0;                                  % DIRECTORY NAME
110075  0;0;0;0;0;0;0;0
110105  0;0
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107
110107  ENDHD=*                                    % END OF PART OF DIRECTORY TABLE DEALING WITH HARD DISKS.
```

```
110107
110107  "8F1U0
110107  DDRTB 1145,FLOP1,S1150,SX1D
110107  0; 1145; FLOP1; S1150; SX1D;
110114  0;0;0;0;0;0;0;0;                                    % DIRECTORY NAME
110124  0;0;0;0;0;0;0;0
110134  0;0
110136
110136  "
110136  ENDDD=*                                  % END OF PART OF DIRECTORY TABLE FOR DISKS.
110136
110136  "8M1U0
110136  MDRTB 560,MATA1,S561,S40D
110136  20000; 560; MATA1; S561; S40D; 0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;
110165
110165  "
110165
110165  ENDDT=*                                          % END OF DIRECTORY TABLE: BOTH DISKS AND MAG TAPE
110165  )KILL GLOB D10M1 TRUM1 TRUM2 D10M2 D33M1 FLOP1 FLOP2
110165  )KILL D66M1 D75M1 D2881 D33M2 D66M2 D75M2 D2882 D38M1 D38M2 D30M1
110165  )KILL D60M1 D90M1 D30M2 D60M2 D90M2 D3751 D3752 D2751 D2752 D23M1 D23M2
110165  )KILL D14M1 D14M2 D20M1 D20M2 D75M3 D75M4 D2883 D2884 D45M1 D45M2
110165  )KILL D471R D472R D473R D474R D70M1 D70M2 D70M3 D70M4
110165  )KILL D471F D472F D473F D474F D281F D282F D283F D284F
110165  )KILL D671F D672F D673F D674F D281S D282S D283S D284S
110165  )KILL D671S D672S D673S D674S
110165  )KILL D271F D272F D273F D274F D271S D272S D273S D274S
110165  )KILL D141F D142F D143F D144F D141S D142S D143S D144S
110165  )KILL D221F D222F D223F D224F D451F D452F D453F D454F
110165  )KILL D221S D222S D223S D224S
110165
110165  %=============================================================================
110165  % 29.15       B I T   F I L E   B U F F E R S
110165  %
110165  BINDX=0
110165  BFBUF=*
110165
110165  "3B1U0+6B1U0+7B1U0+2B1U0+4B1U0+2E1U0+5E1U0+8E1U0+GE1U0+HE1U0
110165  FBFBU S1102
110165  BINDX ; -1; S1102; *+20/
110210  XBINX=BINDX+1;)KILL BINDX;BINDX=XBINX;)KILL XBINX
110210
110210  "8F1U0
110210  FBFBU S1151
110210  BINDX ; -1; S1151; *+20/
110233  XBINX=BINDX+1;)KILL BINDX;BINDX=XBINX;)KILL XBINX
110233
110233  "
110233
110233
110233
110233  ENDBF=*
110233  FCRUN,33CPN
110234  M[illegible]=*
110234
110234
110234
110234  CURR/
044642  )KILL CURR
044642
```

```
044642
044642   &RFAC
044642   %  ============================================================================
044642   %                        REMOTE FILE ACCESS SEGMENT RESERVATION TABLE:
044642   RF$GN, DSSNM              % SEGMENT NUMBER OF FIRST DATA SEGMENT
044643   TEMP=0
044643    &RFAC; XYS=TEMP; )KILL TEMP; TEMP=XYS+1; )KILL XYS;
044643   NRFSG, TEMP
044644   RF$TB=*
044644   * TEMP/
044645   )KILL TEMP
044645
044645
```

```
044645  )LINE%
044645  %=============================================================================
044645  % 46.0        P R O C E S S - I O
044645  %
044645
044645            )LINE%   M A C R O S   F O R   N - 10 / N - 100   F I L E   S Y S T E M
044645
044645  % FOR SINTRAN III VERSION
044645
044645  )KILL SDATA FENTR FLEAV ISTCK MINBT MOUTB DATA
044645
044645          )MCDEF SDATA
        A=6; ]
044645
044645          )MCDEF DATA $PAR
        )KILL A; A=$PAR +1; ]
044645
044645          )MCDEF FENTR
        STD I (ASTCK; COPY SL DA; COPY SB DD; SAB A; JPL I (SPUSH; ]
044645
044645          )MCDEF FLEAV
        SAA -A; JMP I (SPOP; )KILL A; ]
044645
044645          )MCDEF ISTCK
        STA I (ASTCK; LDA (STACK; STA I (CSTCK
        SAA 1; STA I (TDVN; LDA I (ASTCK; ]
044645
044645          )MCDEF MINBT
        JPL I (SINBT; ]
044645
044645          )MCDEF MOUTB
        JPL I (SOUTB; ]
```

```
044645  JLINE044645   %
044645   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  S S C O M  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
044645
044645
044645   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
044645   %       N O R D - N E T     R E S I D E N T   R O U T I N E S                    %
044645   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
044645
044645   *COSTA=*
044645   *"CASY1+CASY2+CASY3+CASY4+CASY5+CASY6+CASY7+CASY8+CASY9
"044645   @LIB 8S3C-,
044645   *8CUX=*
044645   *150000/
150000   *SNETS=*
150000   * SNETS;0;0                    % SEGMENT LIMITS
150003   @ELIB
150003   *"8CLI1+8CLI2+8CLI3+8CLI4+8CLI5+8CLI6+8CLI7+8CLI8+8CLI9
"150003    *ENETS=*
150003   *"8CLI1+8CLI2+8CLI3+8CLI4+8CLI5+8CLI6+8CLI7+8CLI8+8CLI9
"150003    @LIB 8S3C-,
150003   *8CUX/)KILL 8CUX
044645   @ELIB
044645   @DEV 1
044645   @DEV (S-S-J)COS-TAD-RES-CODE
```

```
044645
044645   %%%%%%%%%%%%%%%%%%%%  COS-TAD-RES-CODE  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
044645
044645   *"BADAD
"044645   %=============================================================================
044645   %     T * A * D    R * E * S * I * D * E * N * T     P * A * R T
044645   %=============================================================================
044645
044645   % ------------------------------------------------------------------
044645   %   SAVED PIE IN MLVLOC TO RESET CORRECT STATE OF MLEV IN MLVULOC
044645
044645   INTEGER MLVSAV
044646   %------------------------------------------------------------------
044646
044646   %=================================================================
044646   %            M L V L O C
044646   %
044646   % ROUTINE TO DISABLE MONITOR LEVEL. THE ORIGINAL STATE OF MONITOR LEVEL
044646   % IS SAVED IN MLVSAV
044646   % THE ROUTINE RETURNS WITH ION
044646   SUBR MLVLOC
044646   INTEGER AREG
044647   MLVLOC: *IOF
044650           A=:AREG; *TRA PIE
044652           A=:MLVSAV; MLEV; *MCL PIE        % SAVE PIE AND DISABLE MLEV
044655           A:=AREG; *ION; EXIT
044660   RBUS
044661
044661   %=================================================================
044661   %            M L V U L O C
044661   %
044661   % ROUTINE TO ENABLE MONITOR LEVEL IF BIT 3 IS SET IN MLVSAV
044661   SUBR MLVULOC
044661   INTEGER AREG
044662   MLVULOC: *IOF
044663           A=:AREG:=MLVSAV; 0=:MLVSAV
044666           IF A BIT MLEVL THEN              % MONITOR LEVEL MUST BE ENABLED
044670               MLEV; *MST PIE
044672           FI
044672           A:=AREG; EXIT
044674   RBUS
044675
044675   %=============================================================================
044675   %     T * A * D     S * T * A * C * K    R * O * U * T * I * N * E * S
044675   %=============================================================================
044675   %     EACH TAD HAS A STACK AREA IN PHYSICAL MEMORY. THE AREA IS RESERVED
044675   %     AT SINTRAN STARTUP. BOTH INPUT AND OUTPUT DATAFIELD (TSTADD) POINTS
044675   %     TO THE STACK.
044675   %
044675   %     GLOBAL VARIABLES:   TSBANK  - MEMORY BANK FOR STACK AREA
044675   %                         TSSIZE  - SIZE OF EACH TAD STACK
044675   %     DATAFIELD VARIABLE: TSTADD  - STACK ADDRESS
044675
044675   %=============================================================================
044675   %      T S T O W F L   T S T D E R R
044675   %
044675   % FATAL ERROR IN TAD STACK. ERRFATAL IS CALLED
044675   % WHEN THE SYSTEM IS IN ERRFATAL THE REGISTERS WILL CONTAIN
044675   % SOME DEBUG INFORMATION:
044675   %
```

```
044675   %       A-REG - TAD STACK ADDRESS
044675   %       T-REG - WHERE THIS ROUTINE IS CALLED FROM
044675   %       D-REG - ORIGINAL L-REG
044675
044675   SUBR TSTOWFL,TSTDERR,TSTEND
044675   TSTOWFL: L=:D; CALL ERRFATAL          % TAD STACK OWERFLOW
044677   TSTDERR: L=:D; CALL ERRFATAL          % TAD STACK DISPLACEMENT ERROR
044701   TSTEND:  L=:D; CALL ERRFATAL          % RETURNING AT START OF STACK
044703   RBUS
044704
044704   %=============================================================================
044704   %       B R S T A C K    X R S T A C K
044704   %
044704   % ROUTINE TO RESET TAD STACK
044704   % BRSTACK: B-REG TAD-DATAFIELD INPUT OR OUTPUT
044704   % XRSTACK: X-REG TAD-DATAFIELD INPUT OR OUTPUT
044704
044704   SUBR BRSTACK,XRSTACK
044704   INTEGER TREG,XREG,AREG,TSTAPO
044710   XRSTACK: *IOF
044711           A=:AREG; X=:XREG; X.TSTADD=:TSTAPO; GO FELLS
044716   BRSTACK: *IOF
044717           A=:AREG; X=:XREG; TSTADD=:TSTAPO
044723   FELLS:  T=:TREG:=TSBANK
044725           X:=TSTAPO; TSSIZE; *STATX          % STACK SIZE
044730           X+1; A:=1; *STATX                  % STACK POINTER
044733           T:=TREG; X:=XREG; A:=AREG; *ION
044737           EXIT
044740   RBUS
044742
044742   %=============================================================================
044742   %       S E T S T A C K
044742   %
044742   % ROUTINE TO CREATE STACK AREA FOR A ROUTINE AND SAVE L-REG
044742   % SIZE OF STACK:     LOCATION AFTER CALL
044742   % TAD-DATAFIELD:     B-REG
044742   % ORGIGINAL L-REG:   D-REG
044742   % NORMAL RETURN:     SKIP OK
044742   % ERROR         :    STACK OWERFLOW, ERRFATAL
044742   SUBR SETSTACK
044742   INTEGER TREG,AREG,DREG,XREG; TRIPLE TADREG=TREG
044746   INTEGER TSTMAX,STPOLD,NSIZE,NEWSTP
044752   SETSTACK: *IOF
044753           TAD=:TADREG; X=:XREG                          % SAVE REGISTERS
044755           T:=TSBANK; X:=TSTADD; *LDATX                  % GET MAX STACK SIZE
044760           A=:TSTMAX; X+1; *LDATX                        % GET OLD STACK POINTER
044763           A=:STPOLD; L=:X
044765           X.SO-=:NSIZE; STPOLD+NSIZE+2=:NEWSTP          % GET SIZE AND FIND NEW
044774           IF A>TSTMAX GO STOWF                          % TEST FOR OWERFLOW
044777           T:=TSBANK; X:=TSTADD+1; NEWSTP; *STATX        % SET NEW STACK POINT
045004           A+TSTADD=:X; STPOLD; *STATX                   % SAVE OLD STACK POINT
045010           X-1; DREG; *STATX                             % SAVE ROUTINES L-REG
045013           TADREG; X:=XREG; *ION                         % RESTORE REGISTERS
045016           EXITA                                         % RETURN
045017   STOWF:  TSTADD; P=:T; GO TSTOWFL     % STACK OWERFLOW
045022   RBUS
045024
045024   %=============================================================================
045024   %       T 1 X E X I    T 2 X E X I
045024   %
```

```
045024   % ROUTINES TO DO EXIT VIA THE STACK
045024   % X-REG MUST POINT TO TAD-DATAFIELD
045024   % T1XEXI: EXIT
045024   % T2XEXI: EXITA
045024
045024   SUBR T1XEXI,T2XEXI
045024   INTEGER RETFL,AREG,XREG,TREG,BREG
045031   T2XEXI: K:=1; GO FELLS
045033   T1XEXI: K:="0"
045034   FELLS: *IOF
045035          A=:AREG; X=:XREG; T=:TREG; X:=:B=:BREG
045042          IF K THEN 1=:RETFL ELSE 0=:RETFL FI
045050          T:=TSBANK; X:=TSTADD+1; *LDATX
045054          IF A=1 GO SEMPT                          % FATAL ERROR STACK EMPTY
045057          X+A-2; T:=TSBANK; *LDATX                 % GET L-REG
045063          A+RETFL=:L; X+1; *LDATX                  % FIND PREVIOUS POINTER
045067          X:=TSTADD+1; *STATX                      % POP STACK
045072          A:=BREG=:B:=AREG; X:=XREG; T:=TREG; *ION
045100          EXIT
045101   SEMPT: TSTADD; P=:T; GO TSTEND
045104   RBUS
045106
045106   %=============================================================================================
045106   %       T 1 B E X I     T 2 B E X I
045106   %
045106   % ROUTINES TO DO EXIT VIA THE STACK
045106   % B-REG MUST POINT TO TAD-DATAFIELD
045106   % T1BEXI: EXIT
045106   % T2BEXI: EXITA
045106
045106   SUBR T1BEXI,T2BEXI
045106   INTEGER RETFL,AREG,XREG,TREG
045112   T2BEXI: K:=1; GO FELLS
045114   T1BEXI: K:="0"
045115   FELLS: *IOF
045116          A=:AREG; X=:XREG; T=:TREG
045121          IF K THEN 1=:RETFL ELSE 0=:RETFL FI
045127          T:=TSBANK; X:=TSTADD+1; *LDATX
045133          IF A=1 GO SEMPT                          % FATAL ERROR STACK EMPTY
045136          X+A-2; T:=TSBANK; *LDATX                 % GET L-REG
045142          A+RETFL=:L; X+1; *LDATX                  % FIND PREVIOUS POINTER
045146          X:=TSTADD+1; *STATX                      % POP STACK
045151          AREG; X:=XREG; T:=TREG; *ION
045155          EXIT
045156   SEMPT: TSTADD; P=:T; GO TSTEND
045161   RBUS
045163
045163   %=============================================================================================
045163   %       S T A X    S T T X
045163   %       S T A B    S T X B    S T T B
045163   %
045163   % ROUTINES TO STORE ONE WORD IN THE STACK
045163   % STAX: STORE A-REG X-RELATIVE
045163   % STTX: STORE T-REG X-RELATIVE
045163   % STAB: STORE A-REG B-RELATIVE
045163   % STXB: STORE X-REG B-RELATIVE
045163   % STTB: STORE T-REG B-RELATIVE
045163   % NORMAL RETURN: SKIP OK
045163   % ERROR RETURN : DISPLACEMENT ERROR, ERRFATAL
045163
```

```
045163   SUBR STAX,STTX,STAB,STXB,STTB
045163   INTEGER TSTAAD,CURSTP,EDISP,CAND,AREG,TREG,XREG
045172   STAX:   *IOF
045173           A=:CAND; GO FELLX
045175   STTX:   *IOF
045176           T=:CAND
045177   FELLX:  A=:AREG; X.TSTADD; GO FELLS
045202   STXB:   *IOF
045203           X=:CAND; GO FELLB
045205   STTB:   *IOF
045206           T=:CAND; GO FELLB
045210   STAB:   *IOF
045211           A=:CAND
045212   FELLB:  A=:AREG; TSTADD
045214   FELLS:  A=:TSTAAD; X=:XREG; T=:TREG
045217           L=:X; A:=X.S0-1=:EDISP                          % FIND DISPLACEMENT
045223           T:=TSBANK; X:=TSTAAD+1; *LDATX                  % FIND STACK POSITION
045227           A+TSTAAD=:CURSTP=:X; *LDATX                     % FIND LEGAL AREA
045233           T:=EDISP; IF X+T<<A OR EDISP>-2 GO DPLER        % CHECK DISPLACEMENT
045243           T:=TSBANK; CAND; *STATX                         % STORE WORD
045246           T:=TREG; X:=XREG; AREG; *ION
045252           EXITA
045253   DPLER:  TSTAAD; P=:T; GO TSTDERR              % DISPLACEMENT ERROR
045256   RBUS
045260
045260   %=============================================================================
045260   %       S T A D B      S T A D X
045260   %
045260   % ROUTINES TO STORE TAD REGISTERS IN THE STACK
045260   % STADB: STORE TAD-REG B-RELATIVE
045260   % STADX: STORE TAD-REG X-RELATIVE
045260   % NORMAL RETURN: SKIP OK
045260   % ERROR RETURN : DISPLACEMENT ERROR, ERRFATAL
045260
045260   SUBR STADB,STADX
045260   INTEGER TSTAAD,CURSTP,EDISP,TREG,AREG,DREG,XREG
045267   TRIPLE TADREG=TREG; DOUBLE TAREG=TREG
045267   STADX:  *IOF
045270           TAD=:TADREG; X.TSTADD; GO FELLS
045273   STADB:  *IOF
045274           TAD=:TADREG; TSTADD
045276   FELLS:  A=:TSTAAD; X=:XREG
045300           L=:X; A:=X.S0-1=:EDISP                          % FIND DISPLACEMENT
045304           T:=TSBANK; X:=TSTAAD+1; *LDATX                  % FIND STACK POSITION
045310           A+TSTAAD=:X; *LDATX                             % FIND LEGAL AREA
045313           T:=EDISP; IF X+T<<A OR EDISP>-4 GO DPLER        % CHECK DISPLACEMENT
045323           T:=TSBANK; TAREG; *STDTX                        % STORE TA-REG
045326           X+2; DREG; *STATX                               % STORE D-REG
045331           TAD:=TADREG; X:=XREG; *ION
045334           EXITA
045335   DPLER:  TSTAAD; P=:T; GO TSTDERR              % DISPLACEMENT ERROR
045340   RBUS
045342
045342   %=============================================================================
045342   %       L D A X    L D T X
045342   %       L D A B    L D X B    L D T B
045342   %
045342   % ROUTINES TO LOAD ONE WORD FROM THE STACK
045342   % LDAX: LOAD A-REG X-RELATIVE
045342   % LDTX: LOAD T-REG X-RELATIVE
```

```
045342   % LDAB: LOAD A-REG B-RELATIVE
045342   % LDXB: LOAD X-REG B-RELATIVE
045342   % LDTB: LOAD T-REG B-RELATIVE
045342   % NORMAL RETURN: SKIP OK
045342   % ERROR RETURN : DISPLACEMENT ERROR, ERRFATAL
045342
045342   SUBR LDAX,LDTX,LDAB,LDXB,LDTB
045342   INTEGER TSTAAD,CURSTP,EDISP,REGFL,AREG,TREG,XREG
045351   LDAX:  *IOF
045352          T=:TREG; X=:XREG; 0=:REGFL; GO FELLX
045356   LDTX:  *IOF
045357          A=:AREG; X=:XREG; 2=:REGFL
045363   FELLX: X.TSTADD; GO FELLS
045365   LDXB:  *IOF
045366          A=:AREG; T=:TREG; 1=:REGFL; GO FELLB
045373   LDTB:  *IOF
045374          A=:AREG; X=:XREG; 2=:REGFL; GO FELLB
045401   LDAB:  *IOF
045402          T=:TREG; X=:XREG; 0=:REGFL
045405   FELLB: TSTADD
045406   FELLS: A=:TSTAAD; L=:X; A:=X.S0-1=:EDISP                    % FIND DISPLACEMENT
045413          T:=TSBANK; X:=TSTAAD+1; *LDATX                       % FIND STACK POSITION
045417          A+TSTAAD=:CURSTP=:X; *LDATX                          % FIND LEGAL AREA
045423          T:=EDISP; IF X+T<<A OR EDISP>-2 GO DPLER             % CHECK DISPLACEMENT
045433          T:=TSBANK; *LDATX                                    % LOAD WORD
045435          X:=REGFL
045436          IF X=0 THEN T:=TREG; X:=XREG; GO RETU FI             % LOAD A-REG
045443          IF X=1 THEN A=:X; AREG; T:=TREG; GO RETU FI          % LOAD X-REG
045452          A=:T; AREG; X:=XREG                                  % LOAD T-REG
045455   RETU:  *ION
045456          EXITA
045457   DPLER: TSTAAD; P=:T; GO TSTDERR              % DISPLACEMENT ERROR
045462   RBUS
045464
045464   %=============================================================================================
045464   %       L T A D B      L T A D X
045464   %
045464   % ROUTINES TO LOAD TAD REGISTERS FROM THE STACK
045464   % LTADB: LOAD TAD-REG B-RELATIVE
045464   % LTADX: LOAD TAD-REG X-RELATIVE
045464   % NORMAL RETURN: SKIP OK
045464   % ERROR RETURN : DISPLACEMENT ERROR, ERRFATAL
045464
045464   SUBR LTADB,LTADX
045464   INTEGER TSTAAD,CURSTP,EDISP,TREG,AREG,DREG,XREG
045473   TRIPLE TADREG=TREG; DOUBLE TAREG=TREG
045473   LTADX: *IOF
045474          X=:XREG; X.TSTADD; GO FELLS
045477   LTADB: *IOF
045500          X=:XREG; TSTADD
045502   FELLS: A=:TSTAAD; L=:X; A:=X.S0-1=:EDISP                    % FIND DISPLACEMENT
045507          T:=TSBANK; X:=TSTAAD+1; *LDATX                       % FIND STACK POSITION
045513          A+TSTAAD=:CURSTP=:X; *LDATX                          % FIND LEGAL AREA
045517          T:=EDISP; IF X+T<<A OR EDISP>-4 GO DPLER             % CHECK DISPLACEMENT
045527          T:=TSBANK; *LDDTX                                    % LOAD AD - TA-REG
045531          AD=:TAREG; X+2; *LDATX                               % LOAD A  - D-REG
045534          A=:DREG; TAD:=TADREG; X:=XREG                        % SET CORRECT REGISTERS
045537   RETU:  *ION
045540          EXITA
045541   DPLER: TSTAAD; P=:T; GO TSTDERR              % DISPLACEMENT ERROR
```

```
045544    RBUS
045546
045546    %=====================================================================
045546    % XX.XX      R S R E S T A R T
045546    %
045546    % ROUTINE TO RESTART USER AFTER RECEIVED RESPONSE. PROGRAM IS ALWAYS
045546    % IN IO-WAIT. ISTATE WAS NOT UPDATED WHEN PROGRAM WAS PUT IN WAITING
045546    % STATE.
045546    SUBR RSRESTART
045546    RSRESTART: X=:B:=RTRES
045550           X.STATUS BZERO 5WAIT=:X.STATUS
045553           GO STUPR
045554    RBUS
045555
045555    %=====================================================================
045555    % XX.XX       R E D O M
045555    %
045555    % ROUTINE TO CAUSE MONITOR CALL TO BE REDONE
045555    % ENTRY:      B = DATAFIELD
045555    %             X = RT-PROGRAM
045555    %             D = WORKING AREA
045555    SUBR REDOM
045555    REDOM: *PON
045556           CALL BRSTACK           % RESET TAD STACK
045557           IF X.RTRES-RTREF=0 THEN CALL WDATA FI
045563           D=:B; ZPREG-1=:ZPREG; GO RETSTUPR
045570    RBUS
045574
045574    %=====================================================================
045574    % XX.XX       I E D C H K
045574    %
045574    % ROUTINE TO CHECK IF INPUT IS DONE WHILE DELAYED ESCAPE ACTION IS ON
045574    % ENTRY: B = INPUT-DATAFIELD
045574    % ERROR: EXIT.  A = ERROR-CODE: INPUT DONE IN DELAYED ESCAPE ACTION
045574    % OK:    SKIP RETURN
045574
045574    SUBR IEDCHK
045574    INTEGER DELMASK:=174377  % ZERO 5ESCLOFF, 5WESC, 5WLOC
045575    IEDCHK: IF FLAGB BIT 5ESCLOFF THEN
045600              IF T:=TYPRING BIT 5BAD THEN          %% B A D  ONLY
045603                IF A BIT 5WESC THEN                %  ESCAPE WAITING
045605                   L=:D; CALL CBRESP; D=:L         %  SEND-RESPONSE
045610                FI                                 %
045610              FI                                   %%
045610              *IOF
045611              FLAGB/\DELMASK=:FLAGB; *ION
045615              A:=TER00; EXIT
045617            FI
045617            A:=0; EXITA
045621    RBUS
045623
045623    %=====================================================================
045623    % XX.XX       B D O U T
045623    %
045623    % START OUTPUT DRIVER ROUTINE FOR TAD
045623    % CALLED WITH PAGING OFF.
045623    % ENTRY: B-REG - TAD OUTPUT DATAFIELD,  RETURN EXIT
045623    SUBR BDOUT
045623    BDOUT: *IOF
045624           B=:A; *IRW LV10B DB              % B = TAD OUTPUT DATAFIELD
```

```
045626          "ACTOUT"; *IRW LV10B DT          % T = DRIVER OUTPUT ROUTINE
045630          "SLV10";  *IRW LV10B DP          % P = LEVEL ACTIVATING POINT
045632          LV10; *MST PID
045634          *PION; POF; EXIT                 % ALLOW DRIVER TO EXECUTE
045637  RBUS
045641
045641  %=============================================================================
045641  % XX.XX       C B E R S P
045641  %
045641  % IF TAD, ROUTINE ACTIVATES DRIVER TO SEND ESCAPE-RESPONSE
045641  % ENTRY: B-REG - INPUT-DATAFIELD,  RETURN EXIT AD1
045641  SUBR CBERSP
045641  CBERSP: IF TYPRING BIT 5BAD THEN
045644            L=:A; CALL BRSTACK; A=:L; *IOF  % RESET TAD-STACK
045650            B=:A; *IRW LV10B DB             % B = TAD INPUT DATAFIELD
045652            "BERESP"; *IRW LV10B DT         % T = DRIVER ESCAPE-RESPONSE
045654            "SLV10";  *IRW LV10B DP         % P = LEVEL ACTIVATING POINT
045656            LV10; *MST PID
045660          FI
045660          *PION; EXIT AD1
045662  RBUS
045666
045666  %=============================================================================
045666  % XX.XX       C X R E S P
045666  %
045666  % IF TAD, ROUTINE ACTIVATES DRIVER TO SEND ESCAPE-RESPONSE
045666  % ENTRY: X-REG - INPUT-DATAFIELD,  RETURN EXIT AD1
045666  % ROUTINE IS CALLED WITH INTERUPTS OFF PAGING ON
045666  SUBR CXRESP
045666  CXRESP: L+1=:D
045670           IF X.TYPRING BIT 5BAD THEN
045673             CALL XRSTACK; *IOF            % RESET TAD STACK
045675             X=:A; *IRW LV10B DB           % B = TAD INPUT DATAFIELD
045677             "BERESP"; *IRW LV10B DT       % T = DRIVER ESCAPE-RESPONSE
045701             "SLV10";  *IRW LV10B DP       % P = LEVEL ACTIVATING POINT
045703             LV10; *MST PID; ION; IOF
045707           FI
045707           D=:P
045710  RBUS
045714
045714  %=============================================================================
045714  % XX.XX       B D T O U
045714  %
045714  % OUTPUT TIMEOUT ROUTINE FOR TAD, CALLED WITH INTERUPT OFF
045714  % ROUTINE SENDS OUTPUT BUFFER AT TIMEOUT
045714  % B = TAD OUTPUT DATAFIELD
045714  SUBR BDTOU
045714  INTEGER POINTER LREG
045715  BDTOU: L=:D; CALL MLVLOC; CALL BRSTACK; CALL SETSTACK(0); *POF
045723         CALL SNDBUF; 0/\0; TTMR=:TMR; *PON
045730         CALL MLVULOC; GO T1BEXI
045732  RBUS
045740
045740  %=============================================================================
045740  % XX.XX       C T I B A D
045740  %
045740  % INPUT IOSET ROUTINE FOR TAD
045740  % ENTRY:  B = INPUT-DATAFIELD
045740  %         X = WORKING-AREA
045740  SUBR CTIBAD
```

```
045740    DISP -4; INTEGER XREG,TREG,AREG,DREG; TRIPLE TADREG=TREG; PSID
045740    CTIBAD: L=:D; CALL MLVLOC
045742            CALL BRSTACK; CALL SETSTACK(XREG)
045745            CALL STADB(TADREG); CALL STXB(XREG)
045751            IF X:=PORTNO=0 GO RETU; *POF
045754            IF A=-1 THEN                        % CLEAR BUFFER
045757               CALL CLIDAT; 0/\0
045761               IF A=1 THEN
045764                  CALL CTRMES; 0/\0
045766               FI
045766               DFOPP=:B
045770    NXBFF:     7RESE; T:=0; CALL CREMES; GO ECHEC
045774               CALL SNDBUF; GO ORET; DFOPP=:B; GO RETW          % SEND RESET
046001    ECHEC:     IF A=2 THEN
046004                  IF DFOPP.BUFFID><0 THEN
046007                     CALL MOVITO; GO NXBFF
046011                  FI
046011               FI; GO ORET
046012    RETW:      *IOF
046013               RTREF.STATUS BONE 5WAIT=:X.STATUS
046017               7RECO=:DFOPP.RSPNUM; 1=:MTOR; GO RETU
046025    ORET:      DFOPP=:B; GO RETU
046030            FI
046030    RETU:   CALL LTADB(TADREG); CALL LDXB(XREG); *PON
046035            CALL MLVULOC; GO T1BEXI
046037    RBUS
046057
046057    %=========================================================================
046057    % XX.XX       C T O B A D
046057    %
046057    % OUTPUT IOSET ROUTINE FOR TAD
046057    % ENTRY: B = OUTPUT-DATAFIELD
046057    %        X = WORKING-AREA
046057    %        A = CONTROLL
046057    %        D = PARAMETER
046057    % CALLED WITH INTERUPT OFF
046057    SUBR CTOBAD
046057    DISP -4; INTEGER XREG,TREG,AREG,DREG; TRIPLE TADREG=TREG; PSID
046057    CTOBAD: D=:T:=L; CALL MLVLOC
046062            CALL BRSTACK; CALL SETSTACK(XREG)
046065            CALL STADB(TADREG); CALL STXB(XREG)
046071            IF X:=DFOPP.PORTNO=0 GO RETU; *POF
046075            IF A=-1 THEN                              % CLEAR BUFFER
046100               CALL CLODAT; 0/\0
046102               IF A=1 THEN
046105                  CALL SNDBUF; 0/\0; GO RETU
046110               ELSE
046111                  X:=BUFFID; AD:=TDTADD; 0=:BUFFID=:REMSIZ; CALL PUTPOOL
046116                  GO RETU
046117               FI
046117            FI
046117            IF A=-2 THEN
046122               CALL SNDBUF; 0/\0; GO RETU             % SEND BUFFER
046125            FI
046125            IF A=23 OR A=24 THEN                      % SYSTEM/USER CONTROLL
046133    NWBUF:     IF BUFFID=0 THEN
046135                  CALL GETPOOL; GO NYTRY              % GET NEW BUFFER
046137               FI
046137               CALL LDAB(AREG); IF A=23 THEN 7SYCN ELSE 7USCN FI
046147               T:=2; CALL CREMES; GO ECHK
```

```
046152              CALL LDAB(TREG); CALL WORDPUT; GO RETU; CALL LDAB(AREG)
046160              IF A=23 THEN
046163                 CALL LDAB(TREG)
046165                 IF A=1 OR A=13 OR A=17 THEN CALL SNDBUF; 0/\0 FI
046200              ELSE
046201                 CALL SNDBUF; 0/\0; *IOF
046204                 RTREF.STATUS BONE 5WAIT=:X.STATUS
046210                 7ERRS=:RSPNUM; 1=:MTOR
046214              FI
046214              GO RETU
046215           FI
046215   RETU:   CALL LTADB(TADREG); CALL LDXB(XREG); *PON
046222           CALL MLVULOC; GO T1BEXI
046224
046224   ECHK:   IF A=1 GO NYTRY
046227           IF A=2 THEN
046232              CALL SNDBUF; GO RETU; GO NWBUF
046235           FI
046235           GO RETU
046236
046236   NYTRY:  CALL LDAB(XREG); A=:D; RTREF=:X; CALL MLVULOC; GO REDOM
046245   RBUS
046271
046271   %=========================================================================
046271   % XX.XX       B B R E C
046271   %
046271   % ROUTINE TO SEND BREAK AND ECHO MESSAGE FOR TAD
046271   % ENTRY: X = INPUT DATAFIELD
046271   %         D = CONTROLL (1=BREAK, 2=ECHO)
046271   SUBR BBREC
046271   BBREC:  A:=L+1=:X."LRSA"; IF X.PORTNO=0 GO RETU
046275           CALL XRSTACK; CALL MLVLOC; ZAREG/\377; *POF
046302           IF D=1 THEN CALL BDBREA ELSE CALL BDECHO FI; *PON
046311           CALL MLVULOC; *ION
046313   RETU:   T:=X."LRSA"=:L; EXIT
046316   RBUS
046324
046324   %=========================================================================
046324   % XX.XX       B S T T Y
046324   %
046324   % ROUTINE TO SEND TERMINAL TYPE CALLED FROM MSSTY
046324   % X = INPUT-DATAFIELD
046324
046324   SUBR BSTTY
046324   BSTTY:  A:=L+1=:X."LRSA"
046326           IF X.PORTNO=0 THEN EXITA FI
046331           X:=:B=:XRSA; DFOPP=:B
046335           CALL MLVLOC; CALL BRSTACK; *POF
046340   NXBUF:  7TTYP; T:=2; CALL CREMES; GO ECHEC
046344           DFOPP.CTTYP; CALL WORDPUT; GO RETU
046350   RETU:   *PON
046351           CALL MLVULOC; *ION
046353           DFOPP=:X; X.XRSA=:B; X."LRSA"=:L; EXIT
046362
046362   ECHEC:  IF A=2 THEN CALL SNDBUF; 0/\0; GO NXBUF; FI
046370           GO RETU
046371   RBUS
046377
046377   %=========================================================================
046377   % XX.XX       B T M O D
```

```
046377   %
046377   % ROUTINE TO SEND TERMINAL-MODE, CALLED FROM MTERMODE
046377   %
046377   % B = OUTPUT-DATAFIELD, A = MODE
046377
046377   SUBR BTMOD
046377   BTMOD: T:=L+1=:"LRSA"; IF T:=DFOPP.PORTNO=0 THEN EXITA FI; *POF
046407          A=:XRSA; CALL MLVLOC; CALL BRSTACK                    % RESET TAD STACK
046412   NXBUF: 7TMOD; T:=1; CALL CREMES; GO ECHEC
046416          A:=XRSA/\377; CALL BYTPUT; GO RETU
046422          CALL SNDBUF; 0/\0
046424   RETU:  *PON
046425          CALL MLVULOC; *ION
046427          GO LRSA
046430   ECHEC: IF A=2 THEN CALL SNDBUF; 0/\0; GO NXBUF; FI
046436          GO RETU
046437   RBUS
046446
046446   %-=============================================================================
046446   % XX.XX       C T M O D
046446   %
046446   % ROUTINE TO SEND TERMINAL-MODE, CALLED FROM TERMODE COMMAND
046446   %
046446   % X = INPUT-DATAFIELD
046446
046446   SUBR CTMOD
046446   CTMOD: T:=L+1=:X."LRSA"; IF X.PORTNO=0 THEN EXITA FI; CALL MLVLOC; *POF
046455          X:=:B=:XRSA; DFOPP=:B; CALL BRSTACK                   % RESET TAD STACK
046462   NXBUF: 7TMOD; T:=1; CALL CREMES; GO ECHEC; DFOPP=:B; A:=0
046471
046471   % CAPITAL LETTERS
046471          IF T:=DFLAG BIT 5CAPITAL THEN A BONE "0" FI
046475   % CR DELAY
046475          IF T:=TYPRING BIT 5CRDLY THEN A BONE 1 FI
046501   % STOP ON FULL PAGE
046501          IF T:=DFOPP.SCREEN><X:=0 THEN A BONE 2 FI
046507   % LOGGOUT ON MISSING CARRIER
046507          IF T:=FLAGB BIT 5LBLOG   THEN A BONE 3 FI
046513          X:=DFOPP=:B; CALL BYTPUT; GO RETU
046517          CALL SNDBUF; 0/\0
046521   RETU:  *PON
046522          CALL MLVULOC; *ION
046524          DFOPP=:X; X.XRSA=:B; X."LRSA"=:L; EXIT
046533   ECHEC: IF A=2 THEN CALL SNDBUF; 0/\0; GO NXBUF; FI
046541          GO RETU
046542   RBUS
046550
046550   %-=============================================================================
046550   % XX.XX       B C E S C
046550   %
046550   % ROUTINE CALLED FROM MCDESCFU/MCEESCFU TO SEND ENABLE/DISABLE ESCAPE
046550   % X = INPUT DATAFIELD
046550   SUBR BCESC
046550   BCESC: IF X.PORTNO=0 THEN EXITA FI
046553          A:=L+1=:X."LRSA"; X:=:B=:XRSA; DFOPP=:B
046561          CALL MLVLOC; CALL BRSTACK; *POF               % RESET TAD STACK
046604   NXBUF: 7CESC; T:=1; CALL CREMES; GO ECHECK
046570          IF DFOPP.DFLAG BIT 5IESC THEN A:=0 ELSE A:=1 FI
046577          CALL BYTPUT; GO RETU; CALL SNDBUF; GO RETU; *IOF
046604          RTREF.STATUS BONE 5WAIT=:X.STATUS
```

```
046610          7CERS=:RSPNUM; 1=:MTOR
046614  RETU:   *PON
046615          CALL MLVULOC; *ION
046617          DFOPP=:X; X.XRSA=:B; X."LRSA"=:L; EXIT
046626  ECHEC:  IF A=2 THEN CALL SNDBUF; 0/\0; GO NXBUF; FI
046634          GO RETU
046635  RBUS
046645
046645  %=======================================================================
046645  % XX.XX       B S D A E
046645  %
046645  % ROUTINE CALLED FROM MSDAE TO SEND CHANGE ESCAPE CHARECTER
046645  % X = INPUT DATAFIELD
046645  SUBR BSDAE
046645  BSDAE:  A:=L+1=:X."LRSA"; IF X.PORTNO=0 THEN EXITA FI
046652          X:=;B=:XRSA; DFOPP=:B; CALL MLVLOC; CALL BRSTACK; *POF
046661  NXBUF:  7DESC; T:=1; CALL CREMES; GO ECHECK
046665          DFOPP.CESCP/\377
046670          CALL BYTPUT; GO RETU; CALL SNDBUF; GO RETU
046674  RETU:   *PON
046675          CALL MLVULOC; *ION
046677          DFOPP=:X; X.XRSA=:B; X."LRSA"=:L; EXIT
046706  ECHEC:  IF A=2 THEN CALL SNDBUF; 0/\0; GO NXBUF; FI
046714          GO RETU
046715  RBUS
046724
046724  %=======================================================================
046724  % XX.XX       B I S I Z    O I S I Z
046724  %
046724  % ROUTINE CALLED FROM IBRSIZ WITH:
046724  % B = INPUT-DATAFIELD, X = WORKING-AREA
046724  % TO FIND NUMBER OF CHARACTERS LEFT TO READ IN A TAD
046724  % IF TAD INPUT-BUFFER IS EMPTY, A REQUEST FOR ISIZE IS SENT
046724  % TO PARTNER.
046724  % ROUTINE IS CALLED WITH INTERUPT PAGING OFF
046724  %
046724  % SKIP-RETURN: A = NUMBER OF CHARACTERS IN BUFFER
046724  %              X = NUMBER OF CHARACTERS INCLUDING FIRST BREAK  (BISIZ ONLY)
046724  % RETURN:      A = ERROR CODE
046724
046724  SUBR BISIZ,OISIZ
046724  BISIZ:  0=;XRSA; GO FELLS           % IBRSIZ (MON 313)
046726  OISIZ:  1=;XRSA                     % ISIZE  (MON 66)
046730  FELLS:  X=;"LRSA"; CALL MLVLOC; CALL BRSTACK; *POF % RESET TAD-STACK
046734          IF BUFFID><0 THEN                          % INPUT BUFFER PRESENT
046736             IF CURMES=7BDAT THEN                    % DATA-MESSAGE PRESENT
046742  CDATA:        A;=REMBYT-
046744                IF A-1=0 THEN GO CMORE ELSE GO RETU FI
046750             ELSE
046751  CMORE:        CALL GETMES; GO ENDCH                % GET NEXT MESSAGE
046753                IF A=X:=7BDAT THEN                   % DATA MESSAGE
046756                   IF T=0 THEN                       % EMPTY
046760                      GO CMORE
046761                   ELSE
046762                      T=:A; GO RETU                  % NUMBER OF BYTES FOUND
046764                   FI
046764                ELSE
046765                   CALL CTRMES; GO EROUT
046767                   IF CURMES=7BDAT THEN GO CDATA ELSE GO CMORE FI
046775             FI;FI; GO CMORE
```

```
046776
046776   ENDCH:     IF A<0 GO EROUT
046777              IF A=3 THEN
047002                 CALL SNDREJ; 0/\0; A:=TER01; GO EROUT
047006              FI
047006           FI
047006
047006   % INPUT-BUFFER IS EMPTY OR NOT PRESENT
047006           DFOPP=:B                          % B = OUTPUT DATAFIELD
047010           IF DFOPP.BUFFID><0 THEN           % RETURN INPUT BUFFER TO POOL
047013              0=:X.BUFFID=:X.REMSIZ=:X.CURMES=:X.NOBDIS
047017              A=:T; AD:=X.TDTADD; T=:X; CALL PUTPOOL
047023           FI
047023
047023   TAGAI:  A:=7ISRQ; T:=0; CALL CREMES; GO BCHEC
047027           CALL SNDBUF; GO EROUT; GO WANSW
047032
047032   BCHEC:  IF A<0 GO EROUT
047033           IF A=2 THEN                       % OUTPUT-BUFFER FULL
047036              CALL SNDBUF; GO EROUT; GO TAGAI
047041           FI
047041           DFOPP=:B; RTREF=:X; "LRSA"=:D; *PON
047050           CALL MLVULOC; *ION
047052           GO REDOM                          % TRY WHEN BUFFERS RETURNED
047053
047053   % WAIT FOR ISIZE RESPONSE FROM OTHER SIDE
047053   WANSW:  *PON
047054           CALL MLVULOC; DFOPP=:B                    % B = INPUT DATAFIELD, IOF FROM ULOC
047057           RTREF.STATUS BONE 5WAIT=:X.STATUS
047063           IF XRSA=0 THEN 7ISRS ELSE 7ISRS BONE 17 FI
047071           A=:DFOPP.RSPNUM; "LRSA"=:B; 1=:MTOR; *ION
047100           MIN ZPREG; GO RET                          % RETURN IN IOWAIT
047102
047102   % CHARACTERS PRESENT IN LOCAL BUFFER, RETURN NUMBER
047102   RETU:   *PON
047103           CALL MLVULOC; *ION
047105           B=:X; T:="LRSA"=:B; X:=X.XRSA;
047111           A=:ZAREG; IF X=0 THEN A=:ZXREG FI
047115           MIN ZPREG; GO RET
047117
047117   % ERROR RETURN
047117   EROUT:  *PON
047120           CALL MLVULOC; *ION
047122           T:="LRSA"=:B; A=:ZAREG; GO RET
047126   RBUS
047144
047144   %=====================================================================
047144   % XX.XX       P I S I Z
047144   %
047144   % ROUTINE CALLED FROM ISIZE MON CALL IF TAD
047144   SUBR PISIZ
047144   PISIZ:  *PIOF
047145           B:=:X; GO OISIZ
047147   RBUS
047150
047150   %=====================================================================
047150   % XX.XX       B O S I Z
047150   %
047150   % ROUTINE CALLED FROM OSIZE IF TAD
047150   % X = OUTPUT-DATAFIELD
```

```
047150   SUBR BOSIZ
047150   BOSIZ: IF X.BUFFID><0 THEN
047152              A:=X.REMSIZ
047153          FI
047153          A=:ZAREG; MIN ZPREG; GO RET
047156   RBUS
047157
047157   %=====================================================================
047157   % XX.XX       B S C P C
047157   %
047157   % ROUTINE TO PREPARE SENDING OF COMPLETION CODE, CALLED FROM UEADM.
047157   % THE ROUTINE MUST CHECK DATA TO DETERMINE IF THE CURRENT OPERATION
047157   % OF UEADM IS WRITE COMPLETION CODE.
047157
047157   SUBR BSCPC
047157   DISP -177; INTEGER T2SEG,WPNT; PSID
047157   DISP -165; INTEGER ARRAY POINTER SYARR; PSID
047157   SYMBOL FPART=3,CBC01,CBC02,LPART,AREND=27
047157   INTEGER POINTER TTIF:=TTIFIELD
047160   BSCPC: A:=L+1=:TTIF."LRSA"; IF X.PORTNO=0 THEN EXITA FI
047166          CALL XRSTACK                              % RESET TAD STACK
047167          WPNT=:D
047171          FOR X:=0 TO FPART DO; X:=:D               % CHANGE IN UEDAT BEFORE COMP.CODE?
047176              T:=T2SEG; CALL GET1L; GO ERR; X+1:=:D
047203              IF A><SYARR(X) GO NOCOD
047206          OD; A:=D+2=:D
047213          FOR X:=LPART TO AREND DO; X:=:D           % CHANGE IN UEDAT AFTER COMP.CODE?
047220              T:=T2SEG; CALL GET1L; GO ERR; X+1:=:D
047225              IF A><SYARR(X) GO NOCOD
047230          OD
047232          T:=T2SEG; X:=WPNT+CBC02; CALL GET1L; GO ERR     % COLLECT COMPLETION CODE
047237          A=:D; X-1; CALL GET1L; GO ERR; X:=TTIF:=:B      % --------- " ----------
047245          CALL MLVLOC; *POF
047247          CALL SNDCP; *PON                          % SEND COMPLETION CODE
047251          CALL MLVULOC; X:=:B; *ION
047254   RETU:  X."LRSA"=:L; EXIT
047257   ERR:
047257   NOCOD: X:=TTIF; GO RETU
047261   RBUS
047266
047266
047266   *"-BADAD
"047266n  @DEV 1
047266   @DEV (S-S-J)SINB-X
```

```
047266    %
047266    %=============================================================================
047266    %       S I N B - X
047266    %=============================================================================
047266
047266    %=============================================================================
047266    % 12.1          E R R M O N
047266
047266    %MONITOR CALL FOR USER ERRORS, APPL. LEVEL
047266    %A=ERROR NO.,#DD, T=ADDRESS
047266    SUBR ERRMON
047266    INTEGER MERRNO=?
047266    ERRMON: CALL GET0
047267            IF ZAREG><#35 AND ><#90 AND ><#91 AND <#50 OR >#69 THEN #41 FI
047310            A=:MERRNO; ZTREG; CALL 9ERRA
047313    INTEGER MERRNO
047314            GO RET
047315    RBUS
047326
047326
047326    %=============================================================================
047326    % 4.12          G T R T
047326
047326    % MONITOR CALL: IRT=GETRT(0)
047326    SUBR GTRT
047326    GTRT:   CALL GET0; CURPROG=:ZAREG; GO RET
047332    RBUS
047335
047335
047335    %=============================================================================
047335    % 4.13          R T O N   R T O F F
047335
047335    % MONITOR CALLS TO ALLOW OR INHIBIT RT-ROGRAMS
047335    % CALL RTON(PROG)
047335    % CALL RTOFF(PROG)
047335    SUBR RTON,RTOFF
047335    RTON:   CALL GET1; CALL RTCHECK; X.ACTPRI BZERO 5RTOFF; GO RTO
047342    RTOFF: CALL GET1; CALL RTCHECK; X.ACTPRI BONE 5RTOFF
047346    RTO:    A=:X.ACTPRI; GO RET
047350    RBUS
047353
047353    %=============================================================================
047353    % 4.14          S L R M O
047353
047353    % MONITOR CALL TO SET REMOTE OR LOCAL MODE ON FILE ACCESS OPERATIONS
047353    % PARAMETER: AREG: =0 : LOCAL MODE
047353    %                  ><0 : REMOTE MODE
047353    SUBR SLRMO
047353    SLRMO: CALL GET0; % SET FLAG IF SYSTEM IS CONFIGURED WITH REMOTE FILE ACCESS
047354           IF "5FDSZ"><0 THEN ZAREG=:FACFLAG; FI; GO RET  % FACFLAG ON SYS SEGM OR "RT" SEG
047361    RBUS
047365
047365
047365    %=IB CXCPU-,
047365    %=IB CXCPU
047365    %=============================================================================
047365    %               F X T A D D R
047365    %
047365    % FIND PHYSICAL ADDR OF DATAFIELD
047365    % ENTRY:        X=ADDR OF DATAFIELD (IN RESIDENT)
```

```
047365   %             T=DISPLACEMENT IN DATAFIELD
047365   %
047365   % EXIT:       XT=PHYSICAL ADDR OF LOCATION IN DATAFIELD
047365   %
047365   SUBR FXTADDR
047365   FXTADDR: IF X.TYPRING BIT 5TERM THEN
047370                X:=X.TDFLGADDR; T+X; X:=TDFBANK
047373            ELSE
047374                T+X; X:=0
047376            FI; EXIT
047377   RBUS
047400
047400   %=============================================================================
047400   %             C H D F P A G E
047400   %
047400   % SUBROUTINE TO CHECK IF THE "TERMINAL DATAFIELD WINOW PAGE IS IN THE
047400   % PAGE INDEX TABLE, IF NOT, THEN THE WINOW IS SET UP
047400   %
047400   % THIS ROUTINE IS CALLED IN IOF OR WITH THE MONITOR LEVEL DISABLED
047400   %
047400   % ENTRY: X=ADDR OF TERMINAL INPUT DATAFIELD
047400   % EXIT:  AD IS DESTROYED
047400   %
047400   SUBR CHDFPAGE
047400   INTEGER XREG
047401   INTEGER POINTER IPTUBFPAGE:=177000+5UBFPAGE+5UBFPAGE
047402   DOUBLE POINTER DIPTUBFPAGE=IPTUBFPAGE
047402
047402   CHDFPAGE: *POF
047403           IF "5UBFPAGE*2000"<<=X GO L1
047406           IF X.TYPRING BIT 5TERM THEN
047411   L1:        IF IPTUBFPAGE=0 THEN
047413                X=:XREG; A:=RTREF.WINDOW/\377=:D:=162000; AD=:DIPTUBFPAGE
047422                X:=XREG
047423             FI
047423           FI; *PON
047424           EXIT
047425   RBUS
047431   @ELIB
047431
047431
047431   %=============================================================================
047431   %             P I O F L B Y T   -   P I O F S B Y T
047431   %
047431   SUBR PIOFSBYT,PIOFLBYT
047431   PIOFLBYT: *PIOF; LBYT; PION; EXIT
047435   PIOFSBYT: *PIOF; SBYT; PION; EXIT
047441   RBUS
047441
047441
047441   %=============================================================================
047441   % 11.21          G E T B I T
047441
047441   %SUBROUTINE TO GET BIT FROM BITMAP, ADDRESS IN X, CHAR IN A
047441   %RESULT IN A BIT 17
047441
047441   SUBR GETBIT,SSCGETBIT,VSXGETBIT
047441   INTEGER SHAINSTR(0); *SHA
047442   @LIB CXCPU
047442   SSCGETBIT: *POF
```

```
047443   L1:     A BZERO 7; AD SHZ -4; X+A; SHAINSTR; D SHZ -14+A
047451           X.S0; *EXR SD; PON
047454           EXIT
047455   \\XGETBIT: IF X>>=170000 THEN
047460                  A=:T:=B/\176000:=:X/\1777; X+A; A:=T
047467              ELSE
047470                  *POF
047471              FI; GO L1
047472   @ELIB
047472   @LIB CXCPU-,
047472   GETBIT: A BZERO 7; AD SHZ -4; X+A; SHAINSTR; D SHZ -14+A
047500           X.S0; *EXR SD
047502           EXIT
047503   RBUS
047506
047506
```

```
047506
047506   @LIB CXCPU
047506   %============================================================================
047506   % 2.7   C O P Y   B L O C K
047506
047506   % ( C O P Y B , F L Y T T )
047506
047506   % COPY FROM ONE CORE BLOCK TO ANOTHER CORE BLOCK
047506
047506   % X-REG: ADDRESS OF SOURCE BLOCK
047506   % A-REG: ADDRESS OF DESTINATION BLOCK
047506   % T-REG: NUMBER OF WORDS TO MOVE
047506
047506   SUBR COPYB,FLYTT,PT3COPYB
047506
047506           *SDATA
047506   DISP 0; INTEGER FXREG,FTREG,FAREG,FDREG,FLREG,FBREG; PSID
047506   DISP 0; TRIPLE FLTADRG=FTREG; PSID
047506   DISP 0; INTEGER DISP0; PSID
047506
047506   DISP 6
047506   INTEGER NUMW
047506   INTEGER ADEST
047506   INTEGER 2DESTAD
047506   DOUBLE DDESTAD=ADEST
047506   INTEGER XSOUR
047506   INTEGER 2SOURCAD
047506   DOUBLE DSOURCAD=XSOUR
047506   INTEGER ROUTSWITCH
047506   INTEGER COUNT
047506   INTEGER CACTPRI=COUNT
047506   PSID
047506           *DATA COUNT
047506
047506   ILL:    *JMP *                  % ERROR
047507
047507   % SET USER'S ADDRESS AS CURRENT ALTERNATIVE ADDRESS AREA
047507   CSETAPT: UWLOGADR/\300 SH 1=:T
047513           RTREF.ACTPRI=:CACTPRI
047516           *PIOF
047517           A/\177177\/T=:X.ACTPRI/\3773; *TRR PCR; BSET ZRO
047525           *PION
047526           EXIT
047527   *)FILL
047534   PT3COPYB: *FENTR
047541           CALL SPTOPIT; 1=:ROUTSWITCH
047544           TAD:=FLTADRG; X:=FXREG; GO FELLS
047547
047547   COPYB:  *FENTR                          % NOTE: SAME AS ENTER IN FILE SYSTEM
047554           0=:ROUTSWITCH=:CACTPRI
047556   FELLS:  T=:NUMW; A=:ADEST; X=:XSOURC
047561           IF A SHZ -12=SBFPAGE THEN
047565   % DESTINATION ADDRESS IS "PHYSICAL ADDRESS" (DEVICE BUFFER WINDOW)
047565              FAREG/\1777=:T; A:=RTREF.WINDOW SHZ -10=:D:=0
047575              AD SH 12; D\/T; AD=:DDESTAD
047600              IF FXREG SHZ -12=SUBFPAGE THEN
047605   % SOURCE ADDRESS IS IN USER'S ADDRESS AREA
047605                 CALL CSETAPT
047606                 FXREG/\1777=:D; UWLOGADR/\77 SH 12; D\/A; X:=ADEST
047616                 DO WHILE 4000<<NUMW
```

```
047622                  T-A=:NUMW; L:=A; T:=2DESTAD; *BSET ONE; MOVAP; BSET ZRO
047631                  T=:2DESTAD
047632               OD; L:=T; T:=2DESTAD; *BSET ONE; MOVAP; BSET ZRO
047640               GO FAR CRSAPT
047641            FI
047641            IF A=5BFPAGE GO ILL           % PHYSICAL TO PHYSICAL IS ILLEGAL
047644   % SOURCE ADDRESS IS IN CURRENT LOGICAL ADDRESS AREA
047644            FXREG=:D; X:=ADEST
047647            DO WHILE 4000<<NUMW
047653               T-A=:NUMW; L:=A; T:=2DESTAD; *MOVNP
047660               T=:2DESTAD
047661            OD; L:=T; T:=2DESTAD; *MOVNP
047665            GO OUT; *)FILL
047677         FI
047677
047677         IF FXREG SHZ -12=5BFPAGE THEN
047704   % SOURCE ADDRESS IS "PHYSICAL ADDRESS" (DEVICE BUFFER WINDOW)
047704            FXREG/\1777=:T; A:=RTREF.WINDOW SHZ -10=:D:=0
047714            AD SH 12; D\/T; AD=:DSOURCE
047717            IF FAREG SHZ -12=5UBFPAGE THEN
047724   % DESTINATION ADDRESS IS IN USER'S ADDRESS AREA
047724               CALL FAR CSETAPT
047725               FAREG/\1777=:T; UWLOGADR/\77 SH 12; T\/A; 2SOURCE=:D
047736               DO WHILE 4000<<X:=NUMW
047742                  X-A=:NUMW; L:=A; A:=XSOURC
047746                  *BSET ONE; MOVPA; BSET ZRO
047751                  A=:XSOURC
047752               OD; L:=X; A:=XSOURC
047755               *BSET ONE; MOVPA; BSET ZRO
047760               GO CRSAPT; *)FILL
047767            FI
047767            IF A=5BFPAGE GO FAR ILL       % PHYSICAL TO PHYSICAL IS ILLEGAL
047772   % DESTINATION ADDRESS IS IN CURRENT LOGICAL ADDRESS SPACE
047772            T:=ADEST; 2SOURCE=:D
047775            DO WHILE 4000<<X:=NUMW
050001               X-A=:NUMW; L:=A; A:=XSOURC; *MOVPN
050006               A=:XSOURC
050007            OD; L:=X; A:=XSOURC; *MOVPN
050013            GO OUT
050014         FI
050014
050014   % SOURCE AND DESTINATION ADDRESS ARE IN THE CURRENT LOGICAL ADDRESS AREA
050014         FXREG=:D; T:=FAREG
050017         DO WHILE 4000<<X:=NUMW
050023            X-A=:NUMW; L:=A; *MOVNN
050027         OD; L:=X; *MOVNN
050032         GO OUT
050033
050033   % RESET THE PCR REGISTER
050033   CRSAPT:
050033   OUT:     IF ROUTSWITCH><0 THEN
050035               CALL SPT3PIT
050036            ELSE
050037               IF CACTPRI><0 THEN
050041                  CACTPRI; *PIOF
050043                  A=:RTREF.ACTPRI/\3773; *TRR PCR; PION
050050               FI
050050         FI
050050         *FLEAV
050052
```

```
050052  %=============================================================================
050052  %             F L Y T T
050052  %
050052  % SUBROUTINE TO MOVE 100 WORDS IN THE SAME LOGICAL ADDRESS AREA
050052  %
050052  % ENTRY:      X=SOURCE ADDRESS
050052  %             A=DESTINATION ADDRESS
050052  %
050052  % EXIT:       OK
050052  %
050052  FLYTT: T:=A; D:=X; X:=100; X:=:L; *MOVNN
050057         X=:L:=D-100; EXIT
050063  RBUS
050071
050071  %=============================================================================
050071  %       S E T B F P A G E   X S E T B F P A G E
050071  %
050071  % SUBROUTINE TO SET UP ADDRESS TO DATAFIELD OUTSIDE RESIDENT IN THE
050071  % BUFFER WINDOW
050071  % IF SETBFPAGE:
050071  % ENTRY:      B=ADDRESS OF RESIDENT DATAFIELD
050071  %
050071  % EXIT:       B=LOGICAL ADDRESS, ON PIT3, OF DATAFIELD OUTSIDE RESIDENT
050071  %             RETURNS IN "PIOF" MODE
050071  %
050071  % IF XSETBFPAGE;
050071  % ENTRY:      X=ADDRESS OF RESIDENT DATAFIELD
050071  %
050071  % EXIT:       X=LOGICAL ADDRESS, ON PIT0, OF DATAFIELD OUTSIDE RESIDENT
050071  %
050071  SUBR SETBFPAGE
050071  INTEGER POINTER IPIT0:=177000+5BFPAGE+5BFPAGE; DOUBLE POINTER DPIT0=IPIT0
050072  INTEGER POINTER IPIT3:=177600+L4LGP+L4LGP
050073  DOUBLE POINTER DPIT3=IPIT3
050073  XSETBFPAGE; A:=X.TDFPHPAGE=:D:=162000; *PIOF
050077              AD=:DPIT0; X=:T:=RTREF
050102              X.WINDOW/\377; D SH 10; A+D=:X.WINDOW; X:=T
050110              A:=X.TDFLGADDR/\1777+"5BFPAGE*2000"=:X
050114              *PION; EXIT
050116  SETBFPAGE: A:=TDFPHPAGE=:D:=162000; *PIOF
050122             AD=:DPIT3; A:=3642; *TRR PCR
050125             A:=TDFLGADDR/\1777+"L4LGP*2000"=:B
050131             EXIT
050132  RBUS
050141
050141
050141
050141
050141  %=============================================================================
050141  %             X G T D F A D D R
050141  %
050141  % GET VALUE FROM DATAFIELD
050141  %
050141  % ENTRY:      X=ADDRESS OF RESIDENT DATAFIELD
050141  %             T=DISPLACEMENT IN DATAFIELD
050141  %
050141  % EXIT:       A=VALUE FETCHED FROM THE DATAFIELD
050141  %             T AND D ARE DESTROYED
050141  %
050141  SUBR XGTDFADDR
```

```
050141    XGTDFADDR: IF X.TYPRING BIT 5TERM THEN
050144                  X=:D:=X.TDFLGADDR; X+T; T:=TDFBANK; *LDATX
050151                  X:=D; EXIT
050153               FI; X+T; X.S0; X-T; EXIT
050157    RBUS
050160
050160
050160    %=============================================================================
050160    %              X S T D F A D D R
050160    % PUT VALUE INTO DATAFIELD
050160    %
050160    % ENTRY:      X=ADDRESS OF RESIDENT DATAFIELD
050160    %              T=DISPLACEMENT IN DATAFIELD
050160    %              A=VALUE
050160    %
050160    % EXIT:       T,A AND D ARE DESTROYED
050160    %
050160    SUBR XSTDFADDR
050160    XSTDFADDR: A=:D
050161               IF X.TYPRING BIT 5TERM THEN
050164                  D=:A:=X; X:=X.TDFLGADDR; X+T; T:=TDFBANK; *STATX
050172                  X:=D; EXIT
050174               FI; A:=D; X+T; A=:X.S0; X-T; EXIT
050201    RBUS
050202
050202    %=============================================================================
050202    %              S P T O W I N D O W
050202    %
050202    % SET WINDOW-ADDR TO TERIMINAL ON PIT 0
050202    %
050202    % ENTRY:      B=ADDR OF DATAFIELD
050202    %
050202    % EXIT:       WINDOW IS SET IN PIT 0
050202    %              B=LOGICAL ADDR OF DATAFIELD
050202    %
050202    SUBR SPTOWINDOW
050202    INTEGER POINTER IPIT0:=177000+5BFPAGE+5BFPAGE
050203    DOUBLE POINTER DPIT0=IPIT0
050203    SPTOWINDOW: A:=TDFPHPAGE=:D:=162000; AD=:DPIT0
050207                TDFLGADDR/\1777+"5BFPAGE*2000"=:B
050213                EXIT
050214    RBUS
050217
050217    @ELIB
```

```
050217
050217
050217   %=============================================================================
050217   % 16.12       E D I N B T
050217
050217   % INBT FROM LOG.NO. 0 - EDITED TERMINAL INPUT
050217   SUBR EDINBT
050217   EDINBT: X:="BFIELD":=:B=:EDSVB:=L=:"EDSVL"; "COMSTRING"=:CSTRING
050226           IF RTREF.ACTSEG><OPSEG THEN
050233              IF CPNT=0 OR NCOMPL><0 THEN
050237                 IF X.RSEGM=5RTSG OR A=FRSG1 OR A=FRSG2 THEN 0=:X.RSEGM ELSE A:=0 FI
050254                 A=:EDIRSEGM; T:=OPSEG; CALL MMEXY; T=:EDOSG
050260                 A:=NCOMPL; CALL MONEDIT; GO NORM; GO ERETU
050264   NORM:         0=:CPNT
050265                 IF EDIRSEGM><0 THEN A=:RTREF.RSEGM FI; T:=EDOSG; CALL MMEXY
050273              FI
050273           ELSE
050274              IF CPNT=0 THEN CALL EDIT; 0=:CPNT FI
050300           FI
050300           CALL XCREAD; IF =15 THEN 0=:CPNT FI; CALL SETPARITY
050306           MIN "EDSVL"
050307   RETU:   X:="EDSVL"=:L:=EDSVB=:B; A=:ZAREG; TAD:=ZTADREG; X:=ZXREG; EXIT
050317
050317   % ERROR RETURN FROM MODEDIT ERROR CODE IN D-REG.
050317   % IF NOWAIT RETURN, CALL ADRESS OF UNSUCCSESSFUL ETCI IN A-REG.
050317   % ALL OTHER ERRORS 0 IN A-REG
050317   ERETU: A=:NCOMPL
050320           IF EDIRSEGM><0 THEN A=:RTREF.RSEGM FI
050324           T:=EDOSG; CALL MMEXY; A:=D; GO RETU
050330
050330   % GET ONE CHAR. FROM COMAND BUFFER
050330   XCREAD: X=:D:=CPNT; MIN CPNT; T:=CSTRING; *LBYT
050335           D=:X; EXIT
050337   RBUS
050347
```

```
050347
050347  %-------------------------------------------------------------------------
050347  %       B A C G R O U N D   A L L O C A T I O N   S Y S T E M
050347  %
050347  INTEGER NONTTMCOUNT:=10        % ALLOWED TIME INACTIVE WHEN NOT LOGGED IN (2 MINUTES)
050350  INTEGER TTMWARNING:=144        % TIME INACTIVE BEFORE WARNING IS GIVEN (25 MINUTES)
050351  INTEGER ONTTMCOUNT:=170        % ALLOWED TIME INACTIVE BEFORE LOGGED OUT (30 MINUTES)
050352  INTEGER MBSPRTAB,ASBPRTAB      % PHYSICAL ADDRESS OF SPRTAB
050354  INTEGER AEBPRTAB               % PHYSICAL ADDR OF END OF SBPRTAB
050355  INTEGER AEPRVTTABL             % PHYSICAL ADDR OF END OF PRVTTABLE
050356  INTEGER MBPRVTTABLE=MBSPRTAB,APRVTTABLE  % PHYSICAL ADDRESS OF PRVTTABLE
050357
050357  %-------------------------------------------------------------------------
050357  %       T I M E S L I C E   D E F I N I T I O N S
050357  %
050357  %INTEGER TSLIDLES:=0
050357  INTEGER TSLTUNIT:=14           % BASIC TIME UNITS PER SCHEDULER TIME UNIT
050360  INTEGER TSLSYSPRI:=67          % PRIORITY TO OWNER OF SYSTEM RESOURCE WHICH IS WAITED FOR
050361  INTEGER TSLLOWLG:=40           % LOWEST LEGAL PRIORITY FOR INCREASING PRIORITY WHEN BREAK
050362  INTEGER TSLHTIME:=22           % TIME COUNTER LIMIT FOR HASHING
050363  INTEGER TSLHASHM:=17           % MAX HASH DISPLACEMENT MASK
050364  INTEGER FPTSLICE:=BAK01        % FIRST PROCESS TO TIMESLICE
050365  INTEGER LPTSLICE:=THISS        % LAST PROCESS TO TIMESLICE
050366          DOUBLE DFPTSLICE=FPTSLICE
050366
050366  INTEGER ARRAY TSLBRKELEM:=( 1,10,12,15,21, 1, 1, 1)   % START ELEMENT NUMBER WHEN BREAK
050376  INTEGER ARRAY TSLESCELEM:=( 0, 7,12,15,21, 0, 0, 0)   % START ELEMENT NUMBER WHEN ESCAPE
050406  INTEGER ARRAY TSLLPRITAB:=(24,20,24,24,24,24,24,24)   % HIGHEST PRIORITY ON "ND-500 LOWER TIMESLICE CLASS"
050416
050416  @ICR;
050416  % CHAIN ELEMENT NUMBER    0  1  2  3  4  5  6    7  10 11   12 13 14   15 16 17
050416  %
050416  INTEGER ARRAY TSLPRITAB:=(60,55,44,40,30,24,20,  45,20,16,  46,41,21,  44,34,24,
050436                            17,50,42,35,00,00,00,00,  00,00,00,00,00,00,00,00);
050456  INTEGER ARRAY TSLTIMTAB:=( 1, 3, 6,14,30,50, 4,   4,22,22,   2,22,40,   2,10,40,
050476                            10, 3, 6,14,00,00,00,00,  00,00,00,00,00,00,00,00);
050516  INTEGER ARRAY TSLNEXTAB:=( 1, 2, 3, 4, 5, 6, 5,  10,11,10,  13,14,13,  16,17,20,
050536                            17,22,23, 4,00,00,00,00,  00,00,00,00,00,00,00,00);
050556  @ICR;
050556  INTEGER ARRAY TSLCOUNTA=?
050556  INTEGER ARRAY TSLNTIME=?
050556
050556  %=========================================================================
050556  %             T I M E  S L I C E R
050556  %
050556  % 14.0        S T S L I C E   S T B A C K   A N T I J A M M E R
050556  %
050556
050556  @ICR;
050556  SUBR STSLICE,STBACK,ANTIJAMMER,ANTI2JAMMER,TSTBACK,
050556  S5ESCF,RESAJ,XS5ESC,XSTBACK,GTSLPINDEX;
050556  @ICR;
050556
050556
050556  %-------------------------------------------------------------------------
050556  %             X S T B A C K
050556  %
050556  % SUBROUTINE TO SET BREAK PRIORITY ON TIMESLICED PROCESS
050556  %
050556  % ENTRY:      X=INPUT DATAFIELD ("TERMINAL")
```

```
050556   %
050556   XSTBACK:
050556   @LIB CXCPU
050556           IF X.TYPRING BIT 5TERM THEN X:=X.TDRADDR FI
050562   @ELIB
050562           GO L1
050563
050563   %=============================================================================
050563   %              T S T B A C K   -   S T B A C K
050563   %
050563   % SUBROUTINE TO SET HIGH PRIORITY ON BREAK CHARACTERS
050563   % ENTRY:       B=DATAFIELD
050563   % TSTBACK IS CALLED FROM TERMINAL INPUT/OUTPUT DRIVER
050563   TSTBACK:
050563   @LIB CXCPU
050563           X:=TDRADDR; GO L1
050565   @ELIB
050565   STBACK: X:=B
050566   L1:     IF X.ISTAT><0 THEN                          % PROCESS MUST WAIT FOR I/O ON "TERMINAL"
050570              AD:=DFPTSLICE
050571              IF T:=X.RTRES>>=A AND T<<D THEN
050576                 T-A=:D; A:=0; T:=5RTSIZE; *RDIV ST
050603                 TSLSTATUS(A) BONE 5BRKF=:TSLSTATUS(X)
050607              FI;
050607           FI; EXIT
050610   %=============================================================================
050610   %              X S 5 E S C F
050610   %
050610   % CALLED FROM ESCAPE TO SET HIGH PRIORITY WHEN ESCAPE IS TYPED
050610   % WHILE PROCESS IS IN COMMAND MODE AND ESCAPE IS OFF
050610   %
050610   % ENTRY:       T=BIT TO SET IN FLAGB
050610   %              B=DATAFIELD
050610   %              A=FLAGB
050610   %
050610   % EXIT:        RETURNS TO ROUTINE CALLING ESCAPE
050610   %
050610   XS5ESCF: A\/T=:FLAGB
050612   @LIB CXCPU
050612           IF TYPRING BIT 5TERM THEN X:=TDRADDR ELSE X:=B FI
050620   @ELIB
050620   @LIB CXCPU-,
050620           X:=X.RTRES; GO S5ESCF
050622
050622   %=============================================================================
050622   %              S 5 E S C F
050622   %
050622   % SUBROUTINE TO SET ESCAPE PRIORITY ON TIMESLICED PROCESSES
050622   %
050622   % ENTRY:       X=RT-PROGRAM
050622
050622   S5ESCF: AD:=DFPTSLICE
050623           IF X>>=A AND X<<D THEN
050627              X-A=:D; A:=0; T:=5RTSIZE; *RDIV ST
050634              TSLSTATUS(A) BONE 5ESCF=:TSLSTATUS(X)
050640           FI; EXIT
050641   */FILL
050644
050644   %=============================================================================
050644   %              A N T I J A M M E R   -   A N T I 2 J A M M E R
```

```
050644   %
050644   % SUBROUTINES TO SET HIGH PRIORITY ON THE OWNER OF A SYSTEM RESOURCE
050644   % WHICH IS WAITED FOR BY OTHER PROCESSES.
050644   %
050644   % ENTRY WHEN ANTIJAMMER:      B=DATAFIELD
050644   % ENTRY WHEN ANTI2JAMMER:     X=RT-PROGRAM
050644   %
050644   % EXIT:                       NONE
050644   %
050644   % REGISTERS DESTROYED:        T,A,D,L
050644   %
050644
050644   INTEGER CPROG                % CURRENT PROGRAM ADDRESS
050645   INTEGER B2RG
050646   INTEGER 2XRG
050647   INTEGER C2INDX=?
050647   INTEGER POINTER 2LRG
050650   INTEGER CTYP=B2RG
050650
050650   ANTIJAMMER:
050650           IF TYPRING NBIT RING2 THEN EXIT FI        % RETURN WHEN NOT SYSTEM RESOURCE (ON RING 1)
050654           X=:2XRG:=RTRES; GO FELLS
050657   ANTI2JAMMER: X=:2XRG
050660   FELLS: X=:CPROG; A:=L=:"2LRG"
050663           IF X.STATUS/\377<TSLSYSPRI THEN           % SYSTEM PRIORITY ?
050670              CALL GTSLPINDEX; GO FIN; X=:C2INDX     % FIND TIMESLICE-PROCESS INDEX
050673              IF TSLSTATUS(X) BIT 5NOSLICE THEN
050676                 T:=CPROG.STATUS; X:=77; T/\X; A/\177700\/T   % SAVE ORIGINAL PRIORITY
050704              FI; A BONE 5SPRF=:TSLSTATUS(C2INDX)       % MARK PROCESS IN SYSPRI
050707              CPROG.STATUS/\177400+TSLSYSPRI=:X.STATUS  % SET SYSTEM PRIORITY
050714              CALL FRWQU
050715              IF A><0 THEN A:=:B=:B2RG; CALL TOWQU; B2RG=:B FI % REORGANIZE QUEUE ACCORDING TO PRIORITY
050723   FIN:    FI; X:=2XRG; GO 2LRG; *)FILL
050735
050735   %=============================================================================
050735   %              R E S A J
050735   %
050735   % SUBROUTINE CALLED FROM BRELEASE TO INSERT THE PREVIOUS PRIORITY
050735   % ON THE PROCESS WHICH HAD A SYSTEM RESOURCE THET WAS WAITED FOR.
050735   % (RESAJ MUST BE CALLED BEFORE THE RESOURCE IS RELEASED)
050735   %
050735   % CALLED FROM MONITOR LEVEL OR WITH MONITOR LEVEL DISABLED OR IN IOF
050735   %
050735   % ENTRY:      B=DATAFIELD
050735   % EXIT:       NONE
050735   %
050735   % REGISTERS DESTROYED:        T,A,D,L
050735   %
050735
050735   INTEGER BREG                 % DATAFIELD ADDRESS
050736   INTEGER SNUMPR               % NUMBER OF PROCESSES WAITING FOR THIS DATAFIELD
050737   INTEGER C2INDX               % CURRENT TIMESLICE-PROCESS INDEX
050740   INTEGER POINTER L2RG         % RETURN ADDRESS
050741   INTEGER XREG                 % SAVED X-REG
050742   INTEGER POINTER L3RG
050743
050743   RESAJ:
050743           IF TYPRING BIT RING2 THEN                 % SYSTEM RESOURCE (ON RING 2)
050746              X=:XREG:=L=:"L3RG"
050751              X:=RTRES; CALL GTSLPINDEX; GO RETU; X=:C2INDX      % FIND PROCESS INDEX
```

```
050755              IF TSLSTATUS(X) BIT 5SPRF THEN          % PROCESS HAS ANTIJAM PRIORITY
050760                 T:=0; X:=B+5BWLINK-5WLINK
050764                 DO WHILE X:=X.WLINK><B; T+1; OD      % FIND NUMBER OF PROCESSES WAITING
050771                 T=:SNUMPR
050772                 D:=0; X:=RTRES+5BRESLINK; T:=RTRES
050776                 DO WHILE X:=X.RESLINK><T             % FIND NUMBER OF SYSTEM RESOURCES RESERVED
051001                    IF X.TYPRING BIT RING2 AND X.BWLINK><X THEN D+1 FI
051010                 OD
051011                 A:=RTRES:=:B=:BREG
051014                 IF 1>=D THEN                         % ONLY 1 SYS.RESOURCE, RESET PRIORITY
051017                    TSLSTATUS(C2INDX) BZERO 5SPRF=:TSLSTATUS(X)
051023                    IF A NBIT 5NOSLICE THEN
051025                       X:=37; X/\A; T:=TSLPRITAB(X)  % FIND OLD PRIORITY
051030                    ELSE
051031                       T:=77; T/\A
051033                    FI; STATUS/\177400\/T=:STATUS     % RESET PRIORITY
051037                    X:=B; CALL FRWQU
051041                    IF A><0 THEN B:=A; CALL TOWQU FI % UPDATE QUEUE ACCORDING TO PRIORITY
051044                 FI
051044                 IF SNUMPR>1 THEN                        % MORE THAN 1 PROCESS WAITING?
051050                    X:=BREG.BWLINK; CALL ANTI2JAMMER     % SET ATIJAM PRIOR ON RESERVING PROCESS
051053                 FI; BREG=:B
051055  RETU:       FI; X:=XREG; GO L3RG
051057           FI
051057           EXIT
051060  *)FILL
051067
051067  %============================================================================
051067  %              G T S L P I N D E X
051067  %
051067  % GET TIMELSICE-PROCESS INDEX
051067  %
051067  % ENTRY:            X=RTDESCRIPTION
051067  %
051067  % RETURN:           NOT TIMESLICED PROCESS
051067  % SKIP RETURN:      X=TIMESLICE PROCESS INDEX
051067  %
051067  GTSLPINDEX: AD:=DFPTSLICE
051070           IF X>>=A AND X<<D THEN
051074              X-A=:D; A:=0; T:=5RTSIZE; *RDIV ST
051101              X:=A; EXITA
051103           FI; EXIT
051104
051104  RBUS
051105
051105
051105  %============================================================================
051105  %
051105  %                  M I S C .   R O U T I N E S
051105  %
051105  % 15.0          G B R K D
051105
051105  %MONITOR CALL TO GET BREAKPOINT FOR MACD
051105  % APPL. LEVEL
051105  INTEGER MDSEG:=5ERRSEG\5MACDSEG
051106  INTEGER ARRAY REFBP(10)
051116  SUBR GBRKD
051116  GBRKD: CALL GETO
051117           T:= RTREF.ACTSEG; MDSEG=:X.ACTSEG; "STUPR"; *IRW MLEVB DP
051125           MLEV; *MST PID; MST PIE
```

```
051130          CALL XGBRKD             % ON SYSTEM SEGMENT
051131    RBUS
051136
051136    %=============================================================================
051136    % 15.1         C H L I M
051136
051136    %SUBROUTINE TO CHECK LIMITS FOR MONITOR CALLS
051136    % A=ADDRESS, T=NO. OF WORDS, D=CALLING RT-PROGRAM
051136    % SKIP RETURN IF OK
051136
051136    SUBR CHLIM
051136    INTEGER TREG,AREG,DREG,XREG,XFPAGE,XLPAGE
051144    INTEGER POINTER LREG
051145    REAL TADREG=TREG
051145          % LOCAL SUBROUTINE TO CHECK IF AREA IS WITHIN SEGMENT(IN A)
051145          % RETURN WITH A<0 IF NOT.
051145          XHLIM: X=:D; IF A=0 GO ERRL
051147                 A*5SEGSIZW+SEGSTART; A.LOGADR/\77;  IF >XFPAGE GO ERRL
051157                 T:=X.LOGADR SHZ -10-1+A; IF XLPAGE>T GO ERRL
051166                 "0"
051167    OUT:         D=:X; EXIT
051171    ERRL:        -1; GO OUT
051173
051173    CHLIM: *IOF
051174          TAD=:TADREG; X=:XREG:=L=:"LREG"
051200          IF X:=BACKGROUND=0 THEN                    % NO TEST FOR BACKGROUND
051203             T+A-1 SHZ-12=:XLPAGE; A SHZ-12=:XFPAGE
051211             IF D.STATUS BIT 11 GO OK                % RING 2 OR 3
051215             IF BIT 10 THEN                          %RING 1; TEST CORE COMMON
051217                IF XFPAGE>=CCFPAGE AND XLPAGE<=CCLPAGE GO OK
051227             FI; X.ACTSEG SHZ -10; CALL XHLIM; IF >=0 GO OK
051233             X.ACTSEG/\377; CALL XHLIM; IF>=0 GO OK
051237             X.RSEGM; CALL XHLIM; IF >=0 GO OK
051242             GO ERR
051243          FI
051243    OK:   MIN "LREG"
051244    ERR:  X:=XREG; "LREG"=:L
051247    SKIPRET: TADREG; *ION; EXIT
051252    RBUS
051261
051261    %=============================================================================
051261    % 15.2         M M E X Y
051261
051261    %SUBROUTINE TO CHANGE SEGMENTS WITHOUT USING MONITOR CALL
051261    %T=SEGMENT NUMBERS; RETURN WITH T=OLD SEGMENT NUMBERS
051261    SUBR MMEXY
051261    INTEGER XREG; INTEGER TREG; DOUBLE ADREG; REAL TADREG=TREG
051265    MMEXY: *IOF
051266          TAD=:TADREG; X=:XREG:=RTREF
051271          IF T SHZ -10=A:=377 THEN T:=X.ACTSEG SHZ -10 FI
051277          T SH 10=:D; IF A/\TREG=377 THEN A/\X.ACTSEG FI
051306          T:=X.ACTSEG; A\/D=:X.ACTSEG; "STUPR"; *IRW MLEVB DP
051313          MLEV; *MST PID
051315          X:=XREG; ADREG; *ION; EXIT
051321    RBUS
051324
051324    %=============================================================================
051324    % 15.3         E N T R C O R E
051324
051324    % SUBROUTINE TO ENTER RT AS USER; CALLED FROM ENTRT
```

```
051324   % A=PASSWORD
051324   SUBR ENTRCORE
051324   INTEGER POINTER OFLD:=OFLCK  % OPEN FILE TABLE LOCK FOR RT
051325   INTEGER RTUS:='RT',XREG,RTENFL:=0
051331   INTEGER POINTER LREG
051332   ENTRCORE: X=:XREG:=L=:"LREG"; IF X:=RTENFL><0 THEN MIN "LREG";GO LREG FI
051341           T:="5ERRSEG\377"; CALL MMEXY
051343           OFLD; CALL XLOCK
051345           X:="RTUS"; CALL FILSYS(ENSYS); GO NSKIP; MIN "LREG";MIN RTENFL
051353   NSKIP: OFLD; CALL XUNLOCK
051355           CALL MMEXY; X:=XREG; GO LREG
051360   RBUS
051366
051366   %=============================================================================
051366   % 15.4        B L E V S E T
051366
051366   % SUBROUTINE TO SAVE INBT/OUTBT LEVEL REGISTERS IN DATAFIELD
051366   % CALLED FROM PAGEFAULT - MONITOR LEVEL
051366   SUBR BLEVSET
051366   INTEGER POINTER LREG; INTEGER AREG
051370   BLEVSET: A:=L=:"LREG"; CALL GETDATAFIELD
051373   @LIB CXCPU
051373           IF X.TYPRING BIT 5TERM THEN
051376              CALL CHDFPAGE; X.TDFLGADDR/\1777+"5UBFPAGE*2000"=:X
051403           FI
051403   @ELIB
051403           X+"5BREGBLOCK"; *SRB BLEVB
051405           X-"5BREGBLOCK"; *IRR ALEVB DP
051407           A=:X.DBPREG; X=:D; T:=RTREF.ACTPRI
051413           "100002+ALEVB"=:X.ACTPRI/\3773; *TRR PCR
051417           T=:D.DBACTPRI; "BLEVREACTIVATE"; *IRW ALEVB DP; POF
051424           GO LREG
051425   %SUBROUTINE TO REACTIVATE INBT/OUTBT WHEN PROG. IS RESTARTED
051425   % RT LEVEL
051425           *IOF
051426   BLEVREACTIVATE: *IOF
051427           A=:AREG; "MBL"; *IRW MLEVB DP
051432           MLEV; *MST PID
051434           AREG; *ION
051436   % MONITOR LEVEL:
051436   MBL:    CALL GETDATAFIELD
051437   @LIB CXCPU
051437           IF X.TYPRING BIT 5TERM THEN
051442              CALL CHDFPAGE; X.TDFLGADDR/\1777+"5UBFPAGE*2000"=:X
051447           FI
051447   @ELIB
051447           X+"5BREGBLOCK"; *LRB BLEVB
051451           X-5BREGBLOCK; X.DBPREG; *IRW ALEVB DP
051454           X.DBACTPRI=:RTREF.ACTPRI/\3773; *TRR PCR
051461           BLEV; *MST PID
051463           GO MONEN
051464   RBUS
051476
051476   %=============================================================================
051476   % 15.5        G E T D A T A F I E L D
051476
051476   % SUBROUTINE TO GET ADDRESS TO TERMINAL DATAFIELD
051476   % RETURNED IN X-REG.
051476   SUBR GETDATAFIELD
051476
```

```
051476   GETDATAFIELD;
051476           RTREF-"BAK01"
051500           IF A<0 THEN
051501              IF LGCOLDSTART><0 THEN L=:X; CALL LOGPH; X=:L:=A; EXIT FI
051510              A:=0
051511           FI; A=:D:=0; T:=5RTSIZE; *RDIV ST
051515           IF D><0 THEN CALL ERRFATAL FI
051520   *"-8BACS
"051520             T:=0; X:="XBCKTAB"+A; *LDXTX
051524           EXIT
051525   *"
"051525    RBUS
051533
051533
051533
051533   %=============================================================================
051533   % 15.7          R T L R F I L E   R T L W F I L E   R T L 2 W F I L E
051533   %              T R T L D S E G   R T L R E T
051533   %              L O N A L T       L O F F A L T
051533   %              R T L M C A L L
051533   %
051533   % SUBROUTINES USED BY THE RT-LOADER
051533
051533   SUBR RTLRFILE,RTLWFILE,RTL2WFILE,TRTLDSEG,RTLRET,LONALT,LOFFALT,RTLMCAL
051533
051533   TRIPLE SVTAD=?
051533   INTEGER XXRG=?,LLRG=?,DDRG=?,AARG=?
051533   LOFFALT: *BSET ZRO
051534           TAD=:SVTAD; A:=200; T:=5RT2SG; GO CFELL
051540   LONALT: *BSET ZRO
051541           TAD=:SVTAD; A:=400; T:=5RTFIL
051544   CFELL: A=:D:=L=:LLRG; X=:XXRG
051550           IF RTREF.ACTSEG/\377><T THEN A:=177400; T\/A; CALL MMEXY FI
051560           GO FAR OUT
051561
051561   RTLMCALL: TAD=:SVTAD; X=:XXRG; *BSET ZRO
051564             L.S0+153000=:DDRG; X+1=:LLRG
051572             0=:RTREF.RSEGM; T:=-1; CALL MMEXY; TAD:=SVTAD; K:="0"; *EXR SD
051601             K:=1; A=:AARG; IF K NBIT THEN MIN LLRG FI
051606             5RTSG; CALL MMREENT; GO FAR OUT1
051611
051611   TRTLDSEG: TAD=:SVTAD; X=:XXRG; *PIOF
051614             RTREF.ACTSEG/\377; T:=5RTFIL SH 10; A\/T=:X.ACTSEG
051623             X.ACTPRI/\174177\/1400=:X.ACTPRI/\3773; *TRR PCR
051631             "3MRTL"; *IRW MLEVB DP
051633             MLEV; *MST PID
051635             TAD:=SVTAD; X:=XXRG; *ION; JMP *
051641
051641   3MRTL:    *PON; IRR ALEVB DT
051643   *"-CXCPU
"051643             *IRW ALEVB DP
051644             GO STUPR
051645
051645   RTLRET:   RTREF.ACTSEG/\377=:T:=X.SEGM/\177400\/T=:X.ACTSEG; *PIOF
051656             X.ACTPRI/\174177\/200=:X.ACTPRI/\3773; *TRR PCR
051664             *IRR ALEVB DL; IRW ALEVB DP; PION
051667             GO STUPR
051670
051670   INTEGER CFILNO,CBLCKNO,CNWRD
051673   INTEGER ARRAY CRWPAR:=(CFILNO,"0");INTEGER CMAD
```

```
051676   INTEGER ARRAY 2CRWPAR:=(CBLCKNO,CNWRD)
051700   *MRTRG=*;MRXRE=*+1;MELRT=*+2;MRCPL=*+3
051700   INTEGER TTRG,AARG,DDRG,XXRG,LLRG,RSWITCH
051706   TRIPLE SVTAD=TTRG
051706   *)FILL
051726
051726   RTL2WFILE: X=:XXRG;=0; GO FELL1
051731   RTLWFILE:  X=:XXRG;=1; GO FELL1
051734   RTLRFILE:  X=:XXRG;=2
051736   FELL1:     X=:RSWITCH;=L=:LLRG; TAD=;SVTAD; T=:CFILNO
051743              A.S2=:CMAD; X=:T; X.S3.S0=:CBLCKNO; T.S4.S0=:CNWRD
051756              IF RSWITCH=0 THEN
051760                *BSET ZRO
051761                RTREF.ACTSEG/\177400; T:=5RTFIL\/A; CALL MMEXY
051767                400=:D; CALL DALTON
051772              FI; *BSET ZRO
051773              IF RSWITCH<2 THEN 153120 ELSE 153117 FI; A=:D
052003              T:=CFILNO; "CRWPAR"; *EXR SD
052006              A=:AARG
052007              IF RSWITCH=0 THEN
052011                RTREF.ACTSEG/\177400; T:=5RT2SG\/A; CALL MMEXY
052017   OUT1:      FI; 200=:D
052021   OUT:       CALL DALTON; TAD:=SVTAD; X:=LLRG=:L;=XXRG; EXIT
052027
052027   RBUS
052040
052040
052040   %=============================================================================
052040   % 15.9        E R I O T R A N S
052040
052040   % SUBROUTINE TO READ FROM ERROR DEVICE
052040   % CALLED FROM SIMINBT, OP.SEG.
052040   SUBR ERIOTRANS
052040   ERIOTRANS: L+1=:X; *POF
052043              CALL IOTRANS; X-1; *PON
052046              X=:P
052047   RBUS
052047
052047
052047   %=============================================================================
052047   % 15.10       S E T 5 N O N
052047
052047   % SUBROUTINE TO SET SEGM. 5 NONDEMAND
052047   % CALLED FROM START PROGRAM
052047   SUBR SET5NON
052047   SET5NON: L=:D; T:=1; CALL MMEXY
052052              "SG5".FLAG/\177774=:X.FLAG; CALL MMEXY; D=:P
052060   RBUS
052063
052063
052063   *"8CLI1+8CLI2+8CLI3+8CLI4+8CLI5+8CLI6+8CLI7+8CLI8+8CLI9
"052063
052063   %=============================================================================
052063   % 15.12       S T O B Y T   L O B Y T
052063
052063   % STORE AND LOAD BYTE IN POF
052063   SUBR STOBYT,LOBYT
052063   STOBYT: *PIOF; SBYT; PION; EXIT
052067   LOBYT: *PIOF; LBYT; PION; EXIT
052073   RBUS
```

```
052073
052073    %=============================================================================
052073    % 15.13     S T A 6 X    L D T 6 X    S T A 0 X    S T A 3 X    S T A 4 X    S T A 5 X
052073    %          L D A 1 X    L D A 2 X    L D A 3 X    S T Z 0 X    S T Z 3 X    L D X 0 X
052073
052073    SUBR STA6X,LDT6X,STA0X,STA3X,STA4X,STA5X,LDA1X,LDA2X,LDA3X,STZ0X,STZ3X,LDX0X
052073    STA6X:  *PIOF; STA 6,X; PION; EXIT
052077    LDT6X:  *PIOF; LDT 6,X; PION; EXIT
052103    STA0X:  *PIOF; STA 0,X; PION; EXIT
052107    STA3X:  *PIOF; STA 3,X; PION; EXIT
052113    STA4X:  *PIOF; STA 4,X; PION; EXIT
052117    STA5X:  *PIOF; STA 5,X; PION; EXIT
052123    LDA1X:  *PIOF; LDA 1,X; PION; EXIT
052127    LDA2X:  *PIOF; LDA 2,X; PION; EXIT
052133    LDA3X:  *PIOF; LDA 3,X; PION; EXIT
052137    STZ0X:  *PIOF; STZ 0,X; PION; EXIT
052143    STZ3X:  *PIOF; STZ 3,X; PION; EXIT
052147    LDX0X:  *PIOF; LDX 0,X; PION; EXIT
052153    RBUS
052153
052153    % ============================================================================
052153
052153    % TERMINATE PROGRAM CALLING MON 65 (QERMSG)
052153    SUBR MMLE
052153    MMLE:   *MON 0
052154    RBUS
052154
052154    *'BRFAC
'052154     %=============================================================================
052154    %       T O F E N T R Y
052154    %
052154    % SUBROUTINE TO SET PIT 0 BEFORE CALLING THE FILE-USER SEGMENT
052154    % AND TO RESET TO PIT 3 BEFORE RETURNING TO PT3FUSER
052154    %
052154    SUBR TOFENTRY
052154
052154    TOFENTRY: A:=L=:FENTLREG
052156              CALL SPT0PIT; *JPL I (110000          % 110000 IS START OF FILE-USER SEG.
052160              GO NSKIP; MIN FENTLREG
052162    NSKIP:    A=:AASTCK; CALL SPT3PIT; A:=FENTLREG=:L:=AASTCK
052167              EXIT
052170    RBUS
052175
052175
052175    %=============================================================================
052175    %              F I L U S
052175    %
052175    SUBR FILUS
052175
052175    FILUS:  A=:AASTCK:=L=:FILULREG; CALL SPT3PIT
052201            A:=AASTCK; CALL PT3FUSER; GO NSKIP; MIN FILULREG
052205    NSKIP:  A=:AASTCK; CALL SPT0PIT
052207            A:=FILULREG=:L:=AASTCK
052212            EXIT
052213    RBUS
052220
052220
052220    % ============================================================================
052220    %              S V R B L K
052220    %
```

```
052220   % SUBROUTINE TO SAVE THE REGISTER BLOCK
052220   %
052220   % ENTRY:      X=ADDRESS WHERE THE REGISTER BLOCK WILL BE SAVED
052220   %
052220   SUBR SVRBLK
052220   SVRBLK: *PIOF; SRB 10; PION
052223           EXIT
052224   RBUS
052224
052224   *"-8RFAC
"052224
052224   %=============================================================================
052224   %       R T C G P L O C
052224   %
052224   % READ/WRITE ONE LOCATION IN RT-COMMON
052224   % (THE REST OF THIS ROUTINE IS PLACED IN PAGING-OFF AREA)
052224   %
052224   % ENTRY:      T=0: READ; T=1: WRITE
052224   %             A=VALUE WHEN WRITE
052224   %             X=LOGICAL ADDRESS IN RT-COMMON
052224   %
052224   % EXIT:       ERROR, RT-COMMON ADDRESS NOT FOUND
052224   %
052224   % EXIT+1:     A=VALUE WHEN READ
052224   %
052224   SUBR RTCGPLOC
052224
052224   INTEGER LRG
052225   RTCGPLOC: *PIOF
052226           A:=:L=:LRG:=L
052231           CALL PRTCGPLOC; GO ERR; MIN LRG
052234   ERR:    A=:L:=LRG; A:=:L; *PION; EXIT
052241   RBUS
052242
052242   *"8LAMU
"052242   %=============================================================================%
052242   %
052242   %       M L A M U
052242   %       LAMU MONITOR CALL.   DEMFIELD IS RESERVED ON ENTRY
052242   %       THE FUNCTION ARE:
052242   %          FUNC = 1      CREATE LAMU
052242   %          FUNC = 2      DELETE LAMU
052242   %          FUNC = 3      CONNECT LAMU
052242   %          FUNC = 4      DISCONNECT LAMU
052242   %          FUNC = 5      MOVE PAGES FROM LAMU AREA TO NOTHING.
052242   %                        USED FOR AREAS IN SHARED MEMORY THAT ARE USED
052242   %                        FOR SWAPPING BY OTHER CPUS
052242   %          FUNC = 6      MOVE PAGES FROM LAMU AREA TO SWAPPING
052242   %          FUNC = 7      CHANGE LAMU PROTECTION
052242   %          FUNC =10      GET LAMU INFORMATION
052242   SUBR MLAMU
052242   %             LAMU DESCRIPTOR ELEMENT
052242   % DISP 0
052242   %       INTEGER LAMPP          % FIRST PHYSICAL PAGE IN LAMU
052242   %       INTEGER LAMNP          % NUMBER OF PAGES IN LAMU
052242   %       INTEGER LAMPR          % LAMU PROTECTION
052242   % PSID
052242
052242   %             DEMFIELD DATA
052242   DISP 5REG+11
```

```
052242           INTEGER LFUNC, LAMID, LAMSZ, LAMLOG   % PARAMETERS
052242           INTEGER LAMRT=LAMSZ,LAMPT=LAMSZ, LAMPH=LAMLOG
052242           INTEGER PFLFP=LAMID    % PAGES FROM LAMU FIRST PAGE
052242           INTEGER LADPH          % ADDRESS IN USER SEG OF PH. PAGE NO
052242           INTEGER LADID          % ADDRESS IN USER SEG OF LAMID
052242           INTEGER LAIAR          % ADDRESS IN USER SEG OF ARRAY TO RETURN LAMU-INFORMATION
052242           INTEGER LACTMAX        % ADDRESS OF LAST ELEMENT IN ACTIV LAMU TABLE
052242           INTEGER LATFLAG        % =0 TO ASSURE THAT GAPIT WILL WORK
052242           INTEGER CACLE          % ADDRESS OF CURRENT ACTIV LAMU TABLE ELEMENT
052242           INTEGER LC             % LOOP CONTROL
052242           INTEGER LC1            % LOOP CONTROL
052242           INTEGER CLLP           % CURRENT LAST LOGICAL PAGE
052242           INTEGER LEFREE         % INDEX OF FREE ELEMENT IN CURRENT ACT LAMU T.E.
052242                                  % LAMSZ DUMMY IN CONLA
052242           INTEGER LALPP          % LAST PHYS PAGE IN LAMU AREA
052242           INTEGER LA1PH=LACTMAX  % 1.PHYS PAGE IN LAMU
052242           INTEGER LALPH=CACLE    % LAST PHYS PAGE IN LAMU
052242           DOUBLE  LNPRO=LC       % NO OF PAGES AND PROTECTION
052242
052242   PSID
052242   SYMBOL MLAMF=10                % HIHEST LAMU FUNCTION
052242
052242   %       LAMU ERROR MESSAGES
052242   SYMBOL ILLRA=ER170             % ILLEGAL RANGE ON LAMID
052242   SYMBOL LINUSE=ER171            % LAMU IN USE
052242   SYMBOL LAMTF=ER172             % LAMU TABLE FULL
052242   SYMBOL LILRT=ER173             % ILLEGAL LAMRT
052242   SYMBOL MLRT=ER174              % MAX NO OF LAMUS PR RT PROG REACHED
052242   SYMBOL LNCN=ER175              % LAMU NOT CONNECTED
052242   SYMBOL ILLPH=ER176             % NO LAMU AREA BIG ENOUGH
052242   SYMBOL LNDEF=ER177             % LAMU NOT DEFINED
052242   SYMBOL LILLO=ER178             % ILLEGAL LOGICAL PAGE NUMBER
052242   SYMBOL LOGOV=ER179             % LOGICAL LAMU OVERLAP
052242   SYMBOL XNFLAD=174              % NO FREE LAMU AREA DESCRICTOR
052242
052242   INTEGER POINTER PPASS:=PASSTYPE
052243
052243   MLAMU: S3; *BSET ZRO; STA ,X LADPH; BSET ONE
052247          S2; *BSET ZRO; STA ,X LAIAR; BSET ONE
052253          S1; *BSET ZRO; STA ,X LADID; BSET ONE
052257          CALL GET4; IF LFUNC >> MLAMF OR A = 0 THEN 174; GO FAR ERET FI
052267
052267          IF LFUNC ><3 AND >< 4 AND >< 10 THEN
052301            IF BACKGROUND >< 0 THEN                    % SYSTEM OR RT PROG RING 2
052303               IF  PPASS >< 2 THEN 25; GO FAR ERET FI
052311            ELSE
052312               IF OLDPAG NBIT 1 THEN 25; GO FAR ERET FI
052317            FI
052317          FI
052317   @ICR
052317          LFUNC GOSW FAR ERET, CRELA, FAR DELLA, FAR CONLA,
052325          FAR DISLA, FAR LPTNO, FAR LPTSW, FAR PROLA, FAR INFLA
052331   @CR
052331   %                          CREATE LAMU
052331   CRELA: IF LAMID = 0 THEN
052331                1=:LAMID
052331                FOR LAMID TO GNLAMU-1 DO          % SELECT 1. FREE LAMU ID
052331                   CALL GLAPP
052331                   IF A = 0 GO LAFND
052331                OD
```

```
052331              LAMTF; GO FAR ERET
052353          FI
052353    LAFND: CALL CHLID; CALL GLAPP; IF A >< 0 THEN LINUSE; GO FAR ERET FI
052360          CALL CHAREA                              % IS LAMPH INSIDE LAMU AREAS ?
052361    AROK:  CALL GLAPP; LAMPH; *STATX LMPP           % 1. PHYS.PAGE
052364          LAMSZ;  *STATX LMNP                      % NO OF PAGES
052366          140000; *STATX LMPR                      % DEFAULT PROTECTION R W  RING 0
052370          GO FAR MLRET
052371    *)FILL
052413
052413    %                             DELETE LAMU
052413    DELLA: CALL CHLID
052414          A:=SEGSTART-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
052422          A*GNLPRT-1*ALMSZ+LAMACT=:LACTMAX
052427          FOR X:=LAMACT STEP ALMSZ TO LACTMAX DO      % SEARCH TO FIND IF ANY PROGRAM HAS LAMID CONNECTED
052433                T:=LAMBANK; *LDATX LMCN
052435                IF A = LAMID THEN LINUSE; GO FAR ERET FI
052442          OD
052444          CALL GLAPP; IF A = 0 THEN LNDEF; GO FAR ERET FI
052450          A:=0; *STATX LMPP                % OK.  DELETE THE LAMU
052452          GO FAR MLRET
052453    *)FILL
052467
052467    %                             CONNECT TO LAMU
052467    CONLA: CALL CHLID; CALL GLAPP; IF A = 0 THEN LNDEF; GO FAR ERET FI
052474          *LDATX LMNP
052475          IF A+LAMLOG-1=:CLLP >>= 300 OR LAMLOG << 100 THEN LILLO; GO FAR ERET FI
052511          CALL SACLE; 0=:LC; -1=:LEFREE
052515          FOR LC TO GNLPRT-1 DO
052522                CALL GLCAC; *LDDTX LMCN    % A=CONNECTED LAMU, D=LOGICAL PAGE NO
052524                IF A = LAMID GO ECLA                % LAMU IS ALREADY CONNECTED
052527                IF A = 0 THEN
052530                   IF LEFREE = -1 THEN LC=:LEFREE FI
052536                ELSE
052537                   CALL GAPP; *LDATX LMNP     % CHECK FOR LOGICAL OVERLAP
052541                   T:=D+A-1; IF LAMLOG <<= T AND CLLP >>= D THEN LOGOV; GO FAR ERET FI
052554                FI
052554          OD
052560          IF LEFREE >< -1 THEN
052564                CALL GAAC; LAMID; *STATX            % OK.  LAMU IS CONNECTED
052567                LAMLOG; *STATX LMLP
052571          ELSE
052572                MLRT; GO FAR ERET
052574          FI
052574    ECLA:  GO FAR MLRET
052575    *)FILL
052612
052612    %                             DISCONNECT FROM LAMU
052612    DISLA: CALL SACLE; 0=:LC
052614          IF LAMRT = 0 OR A = SEGPROG THEN; *PIOF
052622                CALL CLAMU; *PION
052624          FI
052624          IF LAMID = -1 THEN
052630                FOR LC TO GNLPRT-1 DO
052635                   CALL GLCAC; A:=0; *STATX
052640                OD
052644          ELSE
052645                CALL CHLID
052646                FOR LC TO GNLPRT-1 DO
052653                   CALL GLCAC
```

```
052654                      IF A-LAMID = 0 THEN
052656                         *STATX
052657                         GO DISOK
052660                      FI
052660                   OD
052664                   LNCN; GO FAR ERET
052666             FI
052666   DISOK:    GO FAR MLRET
052667   *)FILL
052700
052700   %         PAGES TO NOTHING & PAGES TO SWAPPING
052700   LPTSW:
052700   LPTNO:    IF PFLFP=:LAMPH = 0 THEN 174; GO FAR ERET FI
052705             A+LAMSZ-1=:CLLP; CALL CHAREA; L:=X
052712             IF PFLFP >< T THEN
052715                   IF A+LAMSZ-1 >< D THEN
052721                      0=:LC                % FIND A FREE LAMU AREA DESCRIPTOR
052722                      FOR LC TO "NINSZ-1" DO A SH 1+"LAMAR"=:X; T:=0; *LDATX
052733                          IF A = 0 GO NEWAR
052734                      OD; XNFLAD; GO FAR ERET % NO FREE AREA DESCRIPTOR
052742   NEWAR:             LC SH 1+"LAMAR"=:X; A:=PFLFP+LAMSZ; T:=0; *STDTX
052752                   FI
052752                   PFLFP-1; X:=L; T:=0; *STATX 10
052757             ELSE
052760                   IF A+LAMSZ-1 >< D THEN
052764                      A+1; T:=0; *STATX
052767                   ELSE
052770                      A:=0=:D=:T; *STDTX
052774                   FI
052774             FI
052774             IF LFUNC = 6 THEN
053000                   PFLFP=:LC
053002                   FOR LC TO CLLP DO
053006                      A SH 2+CORMSTART=:D; SEGSTART+5SEGSIZE=:X; T:=CORMBANK
053015                      MLEV; *MCL PIE
053017                      X.BPAGLINK; X:=:D; *STATX DPAGL
053022                      A:=1; *STATX DPGPR
053024                      A:=X=:D.BPAGLINK
053027                      IF X.SEGLINK = 0 THEN
053031                         A:="BSEGLINK"
053032                         DO WHILE A.SEGLINK >< -1 OD
053040                         A:=D=:X.SEGLINK; -1=:D.SEGLINK
053045                      FI; MLEV; *MST PIE
053047                   OD
053053             FI
053053             GO MLRET
053054   *)FILL
053063   %                          CHANGE LAMU PROTECTION
053063   PROLA:    CALL CHLID; CALL GLAPP; IF A=0 THEN LNDEF;GO FAR ERET FI
053070             LAMPT /\ 163000; * STATX  LMPR
053073             GO MLRET
053074   *)FILL
053101   %                          LAMU INFORMATION
053101   INFLA:    CALL CHLID; CALL GLAPP; IF A=0 THEN LNDEF;GO FAR ERET FI
053106             A =:LA1PH; *LDDTX LMNP
053110             AD=:LNPRO; GO MLRET
053112   *)FILL
053116
053116   MLRET:    ZPREG+1=:ZPREG; A:=0
053122   ERET:     A=:ZAREG; 0=:LATFLAG; X:=B; CALL GAPIT; CALL DALTON; *BSET ZRO
```

```
053130          IF LFUNC = 1 THEN
053134                A:=LAMID; X:=LADID; *BSET ONE; STA ,X 0; BSET ZRO
053141                A:=LAMPH; X:=LADPH; *BSET ONE; STA ,X 0; BSET ZRO
053146          ELSE IF A= 10 THEN
053152                A:=LA1PH; X:=LAIAR; *BSET ONE; STA ,X 0; BSET ZRO
053157                AD:=LNPRO;          *BSET ONE; STD ,X 1; BSET ZRO
053163          FI;FI
053163          GO RETSTUPER
053164  *)FILL
053167
053167
053167  %       SERVICE FUNCTIONS
053167
053167
053167  %       CHECK FOR LEGAL LAMU AREA
053167  %       ON EXIT WITH LAMPH >< 0     T = FIRST, AND D = LAST PAGE OF OLD AREA
053167  CHAREA: 0=:LC
053170          FOR LC TO "NINSZ-1" DO                                  % LAMID OK. THEN CHECK IF PHYSICAL AREA IS OK
053174                A SH 1 + "LAMAR"=:X; T:=0; *LDDTX
053201                IF A=:T >< 0 THEN                                 % AREA IN USE
053203                   IF LAMPH = 0 THEN                              % SELECT PHYS. PAGES NOT IN USE
053205                      A:=D=:LALPP                                 % LAST PHYS. PAGE IN LAMU AREA
053207  L1:                 IF LAMSZ+T-1=:LALPH <<= X:=LALPP THEN % LAMU AREA BIG ENOUGH
053216                         T=:LA1PH; 1=:LC1;
053221                         FOR LC1 TO "GNLAMU-1" DO                 % CHECK THAT NO LAUMU OCCUPIE SELECTED AREA
053225                            L=:D;CALL GAPP;D=:L
053230                            IF A >< 0 THEN
053231                               *LDDTX LMPP
053232                               IF D+A-1 >>=LA1PH AND A <<= LALPH THEN T+1;GO L1 FI
053244                            FI
053244                         OD
053250                         LA1PH=:LAMPH; EXIT                        % OK, FREE AREA FOUND
053253                      FI
053253                   ELSE
053254                      IF A >>= T AND A <<= D THEN
053260                         IF A+LAMSZ-1 <<= D THEN EXIT ELSE GO ARNOK FI
053267                      FI
053267                   FI
063267                FI
053267          OD
053273  ARNOK:  ILLPH; GO FAR ERET
053275  *)FILL
053302
053302  GLCAC:          % GET LC ELEMENT OF CURRENT ACT LAMU TABLE RECORD.
053302          A:=LC
053303  GAAC:   A*ALMSZ+CACLE=:X; T:=LAMBANK; * LDATX
053310          EXIT
053311  GLAPP:          % GET PHYSICAL PAGE OF LAMID. T & X HOLDS THE ADDRESS
053311          A:=LAMID
053312  GAPP:   A*LDTSZ+LAMDT=:X; T:=LAMBANK; *LDATX LMPP
053317          EXIT
053320  CHLID:          % CHECK IF LAMID IS WITHIN LEGAL RANGE
053320          IF LAMID >>= GNLAMU OR A = 0 THEN ILLRA; GO FAR ERET FI; EXIT
053330  SACLE:          % SET CURRENT ACTIVE LAMU TABLE ELEMENT CACLE
053330          IF LAMRT = 0 THEN
053332                A:=RTREF
053333          ELSE
053334                IF A >>= SEGSTART OR A <<= RTSTART THEN
053342  L1:                LILRT; GO FAR ERET
053344                FI
```

```
053344            FI
053344            A-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
053351            A*GNLPRT*ALMSZ+LAMACT=:CACLE; IF D >< 0 GO L3; EXIT
053360
053360
053360    RBUS
053375    *"
"053375
053375    %"=============================================================================
053375    %                C H L A M
053375    %       CHECK IF A LOGICAL PAGE INTERVAL IS WITHIN A LAMU
053375    %       RETURNS WITH INTERRUPT ON
053375    %       ENTRY:      T = FIRST LOGICAL PAGE
053375    %                   D = LAST LOGICAL PAGE
053375    %                   A = RT DESCR
053375    %       EXIT:       NOT WITHIN A CONNECTED LAMU
053375    %       EXIT+1:     OK.    A = PHYSICAL START PAGE OF BUFFER WITHIN LAMU
053375
053375    SUBR CHLAM
053375
053375    *"BLAMU
"053375     INTEGER FPAGE, LPAGE          % FIRST & LAST LOGICAL INTERVAL
053377    INTEGER CACLE                   % CURRENT ACTIVE LAMU ELEMENT
053400    INTEGER SAVX
053401    INTEGER LC                      % LOOP COUNTER
053402    *"
"053402
053402    CHLAM:
053402    *"BLAMU
"053402             *IOF
053403            X=:SAVX; T=:FPAGE:=D=:LPAGE:=5RTSIZE; A-RTSTART=:D:=0; *RDIV ST
053414            A*GNLPRT*ALMSZ+LAMACT=:CACLE; 0=:LC
053421            FOR LC TO GNLPRT-1 DO
053426                  A*ALMSZ+CACLE=:X; T:=LAMBANK; *LDATX
053433                  IF A >< 0 THEN
053434                     A=:D; *LDATX LMLP
053436                     A:=:D; A*LDTSZ+LAMDT=:X; *LDATX LMNP
053443                     A+D-1
053445                     IF T:=LPAGE <<= A AND FPAGE >>= D THEN
053453                        A-D=:D; T:=LAMBANK; *LDATX LMPP
053457                        A+D; L+1; GO CHLRE
053462                     FI
053462                  FI
053462            OD
053466    CHLRE:  X:=LPAGE=:D:=SAVX; *ION
053472    *"
"053472             EXIT
053473    RBUS
053502
053502    %"=============================================================================
053502    %                C O L A M        C L L A M
053502    %       COPY AND CLEAR LAMU
053502            ENTRY COLAM:      A = SOURCE RT PROG
053502    %                         X = DESTINATION RT PROG
053502    %       CLLAM:            A = RT PROGRAM
053502
053502    SUBR COLAM, CLLAM
053502    INTEGER LREG
053503    COLAM:
053503    *"BLAMU
```

```
"053503             A-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
053510             A*GNLPRT*ALMSZ+LAMACT; A:=:X; A-RTSTART=:D:=0; *RDIV ST
053520             A*GNLPRT*ALMSZ+LAMACT=:T; A:=GNLPRT:=:L=:LREG; L+L
053530   @LIB CXCPU
053530             X=:D:=LAMBANK=:A; *MOVPP
053534   @ELIB
053534   @LIB CXCPU-,
053534             LREG=:L
053536   *"
"053536             EXIT
053537   CLLAM:
053537   *"8LAMU
"053537             A-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
053544             A*GNLPRT*ALMSZ+LAMACT=:X; GNLPRT*ALMSZ+X-1=:D:=0; T:=LAMBANK
053557             FOR X TO D DO
053561                  *STATX
053562             OD
053564   *"
"053564             EXIT
053565   RBUS
053572   SUBR LAMDISCONNECT
053572   *"8LAMU
"053572    %===================================================================
053572   % DISCONNECT ALL LAMUS FROM TH CALLING PROGRAM
053572   % CALLED FROM ESCAPE HANDLING ,MON 0  AND BATCH I/O
053572   %
053572   INTEGER ARRAY LPAA:=("4","-1","0","0")
053576   *"
"053576    LAMDISCONNECT:
053576   *"8LAMU
"053576             A:="LPAA"; *MON 2LAMU; MON 2ERMS
053601   *"
"053601    EXIT
053602   RBUS
053606
053606   % MAKE GLOBAL ENTRIES
053606
053606   SUBR ELO3F,EUSE3,DUSE3,ELON3,ELOF3,DLOF3,T207RET,IBR3S,T206RET
053606   RBUS
053606
053606
053606   *"8ESCX
"053606    %===================================================================
053606   %      E L O N   E L O F F   E L O F U   D L O F U   E U S E L   D U S E L
053606
053606   % RESIDENT PART OF THE EXTENDED ESCAPE-HANDLING MONITOR CALLS
053606   % THE MONITOR CALLS IS EXECUTED ON PIT3-SEGMENT.
053606   SUBR ELON,ELOFF,ELOFU,DLOFU,EUSEL,DUSEL,T2P07
053606   ELON:   CALL GET0; CALL SPT3PIT; GO ELON3          % (MON 302)
053611   ELOFF: CALL GET0; CALL SPT3PIT; GO ELOF3          % (MON 303)
053614   ELOFU: CALL GET0; CALL SPT3PIT; GO ELO3F          % (MON 276)
053617   DLOFU: CALL GET0; CALL SPT3PIT; GO DLOF3          % (MON 277)
053622   EUSEL: CALL GET0; CALL SPT3PIT; GO EUSE3          % (MON 300)
053625   DUSEL: CALL GET0; CALL SPT3PIT; GO DUSE3          % (MON 301)
053630   T2P07: CALL CBERSP; 0/\0; GO T207RET              % ESCAPE RESPONSE IN ELON
053634   RBUS
053645
053645   *"8IBRS
"053645    %===================================================================
053645   %            I B R S I Z
```

```
053645    %
053645    % RESIDENT PART OF IBRSIZ MONITOR CALL. CODE ON PIT3-SEGMENT
053645    @LIB CXCPU
053645    SUBR IBRSIZ,T2C06,T2P06
053645    @ELIB
053645    @LIB CXCPU-,
053645    IBRSIZ: CALL GET0; CALL SPT3PIT; GO IBR3S          % (MON 313)
053650    T2C06: *POF
053651    T2P06: GO BISIZ; 0/\0; *PON                        % CALL TO BISIZ FOR TAD
053654           GO T206RET
053655    @LIB CXCPU-,
053655    RBUS
053662    *"
"053662
053662    %"=============================================================================
053662    %              C L R B M A P   B M F R T D   B M T R T D
053662    %              B M O R
053662    %
053662    % ROUTINES TO MANIPULATE WITH THE RT-DESCRIPTION'S REENTRANT BITMAP
053662    %
053662    SUBR CLRBMAP,BMFRTD,BMTRTD,BMOR
053662
053662    % CLRBMAP:    CLEAR THE REENTRANT BITMAP
053662    %       ENTRY:   X=RTDESCRIPTION; INTERRUPT MUST BE OFF OR MLEV DISABLED
053662    %       EXIT:    PAGING IS ON
053662    %
053662    CLRBMAP: *POF
053663           X:=X.RTDLGADDR
053664           0=:X.BITMAP=:X.BITM1=:X.BITM2=:X.BITM3=:X.BITM4=:X.BITM5=:X.BITM6=:X.BITM7
053674           *PON; EXIT
053676
053676    % BMFRTD:     COPY BITMAP FROM RT-DESCRIPTION TO ARRAY POINTED AT BY B-REG
053676    %       ENTRY:   X=RTDESCRIPTION; B=ARRAY ADDR
053676    %       EXIT:    T,A,D,X ARE DESTROYED
053676    %
053676    BMFRTD:
053676    *"CXCPU
"053676             X.RTDLGADDR+5BITMAP=:D; A:=0; X:=10:=:L; T:=B+BXBITMAP; *MOVPN
053707            X=:P
053710    *"-CXCPU
"053710
053710    % BMTRTD:     COPY BITMAP FROM ARRAY POINTED AT BY B-REG TO RT-DESCRIPTION
053710    %       ENTRY:   X=RTDESCRIPTION; B=ARRAY ADDR
053710    %       EXIT:    T,A,D,X ARE DESTROYED
053710    %
053710    BMTRTD:
053710    *"CXCPU
"053710             T:=X.RTDLGADDR+5BITMAP; X:=0; A:=B+BXBITMAP=:D:=10:=:L; *MOVNP
053721            A=:P
053722    *"-CXCPU
"053722
053722    % BMOR:       RT-DESCRIPTION.BITMAP OR SAVED-BITMAP =: RTDESCRIPTION.BITMAP
053722    %       ENTRY: X=RTDESCRIPTION ADDR; B=ARRAY ADDR
053722    %               MUST BE CALLED IN IOF AND ARRAY POINTED TO BY B-REG MUST
053722    %               BE IN MEMORY WHEN BMOR IS CALLED
053722    %       EXIT:  T,A,D,X ARE DESTROYED
053722    %
053722    BMOR:   X:=X.RTDLGADDR+5BITMAP; T:=0
053725            *LDATX 00; ORA XUBIM,B; STATX 00
053730            *LDATX 10; ORA XUBI1,B; STATX 10
```

```
053733          *LDATX 20; ORA XUBI2,B; STATX 20
053736          *LDATX 30; ORA XUBI3,B; STATX 30
053741          *LDATX 40; ORA XUBI4,B; STATX 40
053744          *LDATX 50; ORA XUBI5,B; STATX 50
053747          *LDATX 60; ORA XUBI6,B; STATX 60
053752          *LDATX 70; ORA XUBI7,B; STATX 70
053755          EXIT
053756  RBUS
053756
053756
053756  %=============================================================================
053756  %               C S T S E G
053756  %
053756  % ROUTINE TO CALL STSEG IN "POF-AREA", CALLED FROM MMREENT
053756  %
053756  SUBR CSTSEG
053756  CSTSEG: MLEV; *MCL PIE; POF; ION
053762          X=:A:=L; CALL STSEG; A:=RTREF:=:X; CALL CLRBMAP      % CLRBMAP SETS PAGING ON
053770          A=:L; MLEV; *MST PIE
053773          EXIT
053774  RBUS
053777
053777  %=============================================================================
053777  %               R E E I N   R E E U T
053777  %
053777  % COPY THE REENTRANT BITMAP BETWEEN TWO RT-PROGRAMS
053777  %
053777  % REEIN: COPY BITMAP FROM SSREF TO RTREF
053777  % RREUT: RTREF.BITMAP\/SSREF.BITMAP=:SSREF.BITMAP
053777  %
053777  SUBR REEIN,REEUT
053777
053777  INTEGER LREG,BREG,XREG
054002
054002  REEIN: K:=1; GO FELLS
054004  REEUT: K:="0"
054005  FELLS: *PIOF
054006          X=:XREG:=L=:LREG:=SSREF:=:B=:BREG:=RTREF
054015          IF K THEN
054017             RSEGM=:X.RSEGM
054021             RTDLGADDR+"5BITMAP-BXBITMAP"=:B; CALL BMTRTD
054025          ELSE
054026             0=:X.RSEGM; X:=X.RTDLGADDR+"5BITMAP-BXBITMAP":=:B; CALL BMOR
054033          FI; X:=BREG=:B:=LREG=:L:=XREG; *PION
054041          EXIT
054042  RBUS
054045
054045  *"8MOLI
"054045   %=============================================================================
054045  %
054045  % MON LOGIN
054045  %               <LOG.DEV>       : LOGICAL DEVICE NUMBER OF THE TERMINAL
054045  %               <USER NAME>
054045  %               <PASSWORD>
054045  %               <PROJECT PASSWORD>
054045  %               <SUBSYSTEM NAME>
054045  %               <PASET PARAMETERS>
054045  %               <RETURNED STATUS>
054045  %
054045  SUBR MLOGIN,MLGTMOUT,3MLOGIN
```

```
054045
054045    MLOGIN: CALL GET1; CALL S3NOALPIT; GO 3MLOGIN
054050
054050    % TIMEOUT SUBROUTINE FOR MON LOGIN
054050    MLGTMOUT: -1=:MLIMSTATUS; 0=:TMR
054053              RTRES.STATUS BZERO 5WAIT=:X.STATUS
054057              EXIT
054060    RBUS
054063
054063
054063    %=============================================================================
054063    %       M L I D F I E L D
054063    %
054063    % MONITOR CALL WORKING FIELD FOR MON LOGIN
054063    %
054063    * MLGTM;0;-100;0;0;0
054071    INTEGER ARRAY MLIDFIELD(0)
054071    * 0;0;*-2;0
054075    * *+140/
054235    *"
"054235
054235    %-----------------------------------------------------------------------------
054235    %       S P T 3 P I T   -   S P T 0 P I T   -   S 3 N O A L P I T
054235    %
054235    % SET PAGE INDEX TABLE 0 OR 3 FOR CALLING PROGRAM
054235    %
054235    % SPT3PIT:    NORM PIT=3; ALT PIT=0
054235    % SPT0PIT:    NORM PIT=ALT PIT=0
054235    % S3NOALPIT: NORM PIT=ALT PIT=3
054235    %
054235    SUBR SPT3PIT,SPT0PIT,S3NOALPIT
054235    INTEGER SVX,SVL,CPITS
054240    S3NOALPIT: A:=3600; GO FELLS
054242    SPT0PIT: A:=0; GO FELLS
054244    SPT3PIT: A:=3000
054245    FELLS:    *PIOF
054246              A=:CPITS
054247              A:=L=:SVL; X=:SVX:=RTREF
054253              A:=X.ACTPRI/\174177\/CPITS
054256              A=:X.ACTPRI/\3773; *TRR PCR; BSET ONE
054262              X:=SVX; A:=SVL; *PION
054265              A=:P
054266    RBUS
054273
054273
054273    *"8GPI0+8GPI1+8GPI2+8GPI3+8GPI4+8GPI5+8GPI6+8GPI7
"054273    %=============================================================================
054273    %======== SUBROUTINE TO START ROUTINE ON LEVEL 12 TO CALL DRXMSG =========
054273    %         CALLED FORM GPIB-DRIVER IN POF
054273    %
054273    %-----------------------------------------------------------------------------
054273    SUBR GDRXM
054273     INTEGER 11STS                          % SAVED STATUS REGISTER BEFORE INTERRUPT OFF  LEVEL 11
054274     INTEGER 12STS                          % SAVED STATUS REGISTER BEFORE INTERRUPT OFF  LEVEL 12
054275     TRIPLE  GTREG                          % SAVELOCATION FOR TAD REG
054300     INTEGER GXREG                          % SAVELOCATION FOR X REG
054301     INTEGER DRETA                          % RETURN ADDRESS AFTER CALL TO GDRXM
054302
054302    GDRXM:  TAD=:GTREG;A:=L+1=:DRETA;X=:GXREG
054306            *TRA STS;STA 11STS;IOF                          % TURN OF INTERRUPT
```

```
054311          X:=L;*LDA ,X;STA 12FUN
054314          A:="12DRX";*IRW 140 DT
054316          A:="SLV12";*IRW 140 DP
054320          A:=LV12;*MST PID
054322          *PION
054323          A:=11STS;IF A BIT 17 THEN
054326             *ION                                    % TURN ON INTERRRUPT IF BIT 17 IN STS
054327          FI
054327          A:=DRETA=:L;TAD:=GTREG;X:=GXREG
054333          *POF                                      % TURN OF PAGING BEFORE RETURN TO POF
054334          EXIT
054335
054335    %=============================================================================
054335    %======= ROUTINE ON LEVEL 12 TO CALL DRXMSG ==================================
054335    %=============================================================================
054335
054335    12DRX: *TRA STS;STA 12STS;IOF                     % TURN OF INTERRUPT
054340          X:=GXREG;TAD:=GTREG
054342          CALL DRXMSG                               % GET BUFFER ADDRESS
054343    12FUN: GO 12FUN;GO 12ERR                        % FIRST INSTRUCTION IS DUMMY
054345          TAD=:GTREG;A:=DRETA+1=:DRETA;GO 12RET
054352    12ERR: TAD=:GTREG
054353    12RET: X=:GXREG
054354          T:=12STS;IF T BIT 17 THEN
054357            *ION                                    % TURN ON INTERRRUPT IF BIT 17 IN STS
054360          FI
054360          GO WT12
054361    RBUS
054366
054366    *"8BACS
"054366     %=============================================================================
054366    %       BACKGROUND PROCESS ALLOCATION SYSTEM ROUTINES
054366    %
054366    % START ADDRESS FOR BACKGROUND PROGRAMS
054366    %
054366    % ENTRY:      B=ADDRESS OF TERMINAL (TAD,BATCH) INPUT DATAFIELD
054366    %
054366    SUBR 9ENTOPCOM,MBABPROC,T2P03,9GTLOGDV,GBTINDX,PRLOGOUT,GDBPADDR,CMBABPROC
054366
054366    %=============================================================================
054366    %             C F L O G D V
054366    %
054366    % LOCAL SUBROUTINE TO SEARCH FOR A DATAFIELD ADDRESS
054366    % IN A LOGICAL-DEVICE NUMBER GROUP TABLE
054366    %
054366    % ENTRY: B=ADDRESS OF DATAFIELD
054366    %        X= LOGICAL NUMBER GROUP TO TEST AGAINST
054366    %
054366    % EXIT:  D=LOGICAL DEVICE NUMBER
064366    %
054366    % EXITA: NO LOGICAL DEVICE NUMBER FOUND
054366    %
054366    CFLOGDV: A:=X SH 6=:D; X:=CNVRT(X)
054372             T:=X.S0 SH 1+X+1                       % LAST ENTRY IN LOGICAL DEVICE NUMBER GROUP
054375             X+1
054376             DO
054376                 IF X.S0=B THEN EXIT FI             % LOGICAL DEVICE NUMBER FOUND
054402                 IF X=T THEN EXITA FI               % END OF TABLE
054405                 D+1; X+2
054407             OD
```

```
054410
054410  %=============================================================================
054410  %              9 G T L O G D V
054410  %
054410  % SUBROUTINE TO FIND LOGICAL DEVICE NUMBER
054410  % OF TERMINAL (OR TAD ETC)
054410  % (THIS ROUTINE IS NOT REENTRANT!!)
054410  %
054410  % ENTRY:       B=ADDRESS OF DATAFIELD
054410  %
054410  % EXIT:        NO LOGICAL DEVICE NUMBER FOUND
054410  %
054410  % EXIT+1:      D=LOGICAL DEVICE NUMBER
054410  %
054410  INTEGER POINTER LREG1
054411  9GTLOGDV: A:=L=:"LREG1"; *POF
054414          X:=0; CALL CFLOGDV; GO FOUND
054417          X:=10; CALL CFLOGDV; GO FOUND
054422          X:=12; CALL CFLOGDV; GO FOUND
054425          X:=14; CALL CFLOGDV; GO FOUND
054430          X:=15; CALL CFLOGDV; GO FOUND
054433          X:=20; CALL CFLOGDV; GO FOUND
054436          *PON
054437          GO LREG1
054440  FOUND: MIN "LREG1"; *PON
054442          GO LREG1
054443  *)FILL
054444
054444  %=============================================================================
054444  %              G D B P A D D R
054444  %
054444  % LOCAL SUBROUTINE TO FIND ADDRESS OF AN ELEMENT IN SBPRTAB
054444  %
054444  % ENTRY:       A=PROGRAM
054444  % EXIT:        NO ENTRY FOUND
054444  % EXIT+1:      TX=PHYSICAL ADDRESS OF ENTRY
054444  %
054444  GDBPADDR: X:=ASBPRTAB; A=:D
054446          DO WHILE X<<AEBPRTAB
054451             T:=MBSPRTAB; *LDATX TXSVB
054453             IF A=D THEN EXITA FI
054456             X+BPRTSIZE
054457          OD; EXIT
054461
054461  %=============================================================================
054461  %              9 E N T O P C O M
054461  % START-ADDRESS FOR BACKGROUND PROGRAMS
054461  %
054461  INTEGER POINTER PMXCT:=MAXCT
054462  INTEGER ABPLI:=RTREF
054463  9ENTOPCOM:
054463          IF RTREF>>="2THSS" GO ENTOPCOM
054467          CALL GDBPADDR; GO ERRIN; *LDATX TXBPR  % TEST IF THE SYS.SEG IS INITIALIZED
054472          IF A NBIT BPSOK THEN
054474  ERRIN:     CALL 9ERR(#97); "ABPLI"; *MON 2ABOR % SYS.SEG NOT INITIALIZED
054500          FI; A BZERO BPSOK; *STATX TXBPR
054502          MLEV; *MCL PIE
054504  *IB CXCPU
054504          IF TYPRING BIT 5TERM THEN
054507             % SET UP WINDOW TO TERMINAL DATAFIELD
```

```
054507              T:=TDFPHPAGE; RTREF.WINDOW/\177400\/T=:X.WINDOW
054515           FI
054515   @ELIB
054515           CALL 9GTLOGDV; CALL ERRFATAL                % FIND LOGICAL DEVICE NUMBER OF TERMINAL, TAD, ETC.
054517           MLEV; *MST PIE                              % D=LOGICAL DEVICE NUMBER
054521   @LIB CXCPU
054521           IF TYPRING BIT 5TERM THEN
054524              TDFLGADDR/\1777+"5UBFPAGE*2000"          % ADDR OF TTIFIELD IS IN WINOW (PAGE 77).
054527           ELSE
054530              A:=B                                     % ADDRESS OF TTIFIELD
054531           FI
054531   @ELIB
054531   @LIB CXCPU-,
054531           A:=:D; AD=:"TTNO".DSO                       % UPDATE TTNO AND TTIFIELD
054534           X:=B; CALL GBTINDX; GO ENTOPCOM             % FIND INDEX OF DATAFIELD IN BACKT
054537           A*5PRVTTABLE+APRVTTABLE=:X; T:=MBPRVTTABLE % TX=PHYSICAL ADDRESS OF ELEMENT IN PRVTTABLE
054543           *LDDTX TXUEF                                % UNSAVE SYS.SEG VARIABLES
054544           A=:UEFLG:=D=:PMXCT; *LDDTX TXFLL
054550           A=:FLLIPCOM:=D=:FLBGTERM
054553           GO ENTOPCOM                                 % GO TO "OLD ENTRYPOINT"
054554   *)FILL
054601   %=============================================================================
054601   %              C M B A B P R O C
054601   %
054601   % CALLED FROM MON LOGIN
054601   %
054601   CMBABPROC: CALL MBABPROC; GO MONEN
054603           A:=X; *IRW ALEVB DX; IRR ALEVB DP; AAA 1; IRW ALEVB DP
054610           GO MONEN
054611
054611   %=============================================================================
054611   %              M B A B P R O C
054611   %
054611   % ALLOCATE BACKGROUND PROCESS
054611   %
054611   % MONITOR LEVEL - CALLED FROM MESCAPE
054611   %
054611   % ENTRY:       X= DATAFIELD
054611   %
054611   %
054611   INTEGER POINTER PS41FLG:=S41FLG                      % POINTER TO SG41.FLAG
054612   INTEGER LREG
054613   MBABPROC:
054613           IF X.RTRES=0 THEN                           % INPUT DATAFIELD IS FREE
054615              A:=L=:LREG; X=:B
054620              IF TYPRING /\ BTYPRMASK = 0 THEN  EXIT FI % NOT TERMINAL OR TAD
054624              T:="DFOPP"; CALL XGTDFADDR
054626              IF A><0 THEN                             % TWO WAYS DEVICE
054627   @LIB CXCPU
054627                 IF T:=TYPRING BIT 5TERM THEN A:=B+"9CXTI" FI
054634   @ELIB
054634   @LIB CXCPU-,
054634                 IF A.RTRES=0 AND PS41FLG BIT 5FIX THEN % OUTPUT DATAFIELD IS FREE AND SEGMENT 41 IS FI
054642                    *PIOF; LDA (3632; TRR PCR; PION     % SET NORM.PIT=ALT.PIT=3
054646                    CALL PT3MBAPROG; GO NFOUND          % CONTINUE ON PIT3-SEGMENT
054650                    T:=LREG+1
054652   FELLS:           *PIOF; SAA 32; TRR PCR; PION        % SET NORM.PIT=ALT.PIT=0
054656                    T=:P                                % SKIP RETURN OR RETURN TO MONEN
054657   NFOUND:
054657   T2P08:           CALL CBERSP; 0/\0                   % SEND ESCAPE RESPONSE IF TAD
```

```
054661                      T:="PMONEN"; GO FELLS
054663                   FI
054663                FI; X:=B; LREG=:L
054666             FI; EXIT
054667    *)FILL
054700
054700    %=============================================================================
054700    %              G B T I N D X
054700    %
054700    % SUBROUTINE TO FIND THE BACKGROUND TABLE INDEX OF A DATAFIELD
054700    %
054700    % ENTRY:       X=ADDRESS OF DATAFIELD
054700    %
054700    % EXIT:        NOT FOUND IN BACKT
054700    %
054700    % EXIT+1:      A=INDEX IN BACKT
054700    %
054700    GBTINDX: X=:D:="BACKT"
054702             DO WHILE X.SO><-1
054706                IF A=D THEN A:=X-"BACKT"; X:=D; EXITA FI          % INDEX IS FOUND
054714                X+1
054715             OD; X:=D; EXIT                                       % NOT FOUND IN BACKT
054720
054720
054720    %-----------------------------------------------------------------------------
054720    %              P R L O G O U T
054720    %
054720    % SUBROUTINE TO  CALL STESCAPE, (CALLED FROM 9BPTMOUT ON PIT3-SEGMENT)
054720    % (CALLED IN IOF, RETURNS IN PION)
054720    % (NOT REENTRANT!!)
054720    %
054720    % ENTRY:       B=ADDRESS OF DATAFIELD
054720    %              A=ROUTINE TO CALL (STESCAPE OR ESCAPE)
054720    %
054720    %
054720    INTEGER POINTER PRLLL,CROUT
054722    PRLOGOUT: A=:"CROUT":=L=:"PRLLL"
054725             *POF; SAA 12; TRR PCR
054730             CALL CROUT
054731             *POF; LDA (3612; TRR PCR; PION
054735             GO PRLLL
054736
054736    RBUS
054740    *"-8BACS
"054740     @DEV 1
054740    @DEV (S-S-J)CDR1
```

```
054740
054740   %=============================================================================
054740   %       C D R 1
054740   %=============================================================================
054740
054740
054740   *"MODIN
"054740   %=============================================================================
054740   % 33.1        M O S T I   M O T R I   M O S T O   M O T R O
054740   %             M O T M O   M O T M I
054740   %
054740
054740   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
054740   %     SYNCRONOUS MODEM DRIVER                              %
054740   %     AND HANDLING ROUTINES FOR SINTRAN III    %
054740   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
054740
054740   SUBR MOSTI,MOTRI,MOSTO,MOTRO,MOTMO,MOTMI,TDBPUT,TDBGET
054740   DISP -12
054740   INTEGER MOWSTAT
054740   INTEGER TERSW,MOSW
054740   PSID
054740   % SET STATUS FOR MODEM INPUT
054740   % A=-3: READ MOSW, AND RETURN IT TO THE USER
054740   % A=-2: INITIATE BLOCK RECEIVE (SEARCH FOR SYNC)
054740   % A=-1: CLEAR INPUT BUFFER
054740   % A= 0: UNDEFINED TERMINATE CONDITION
054740   % A= 1: SET DCT-2000 TERMINATE CONDITION
054740   % A= 2: SET CD-200USER TERMINATE CONDITION
054740   % A= 3: SET GERTS-115 TERMINATE CONDITION
054740   % A= 4: SET IBM-3780/2780 TERMINATE CONDITION
054740   % A= 5: SET VIP TERMINATE CONDITION
054740   % A= 6: SET HASP WS TERMINATE CONDITION
054740   % A= 7: SET UNIVAC NTR TERMINATE CONDITION
054740   % A=10: SET MSV2 TERMINATE CONDITION
054740   % A=11: SET IBM-3270 TERMINATE CONDITION
054740   % A=12: SET BTH/VIP7700 TERMINATE CONDITION
054740   % A=13: SET STANSAAB TERMINATE CONDITION
054740   % A=14: SET UTS-400  TERMINATE CONDITION
054740   %
054740   %
054740   %       SET STATUS FOR MODEM INPUT
054740   %
054740   MOSTI: IF A<0 GO MOSI5
054741          IF A>14 GO MOSI6
054744          A=:TERSW; EXIT                    % SET TERMINATE CONDITION
054746   MOSI5: IF A+4<0 GO MOSI6
054750          GOSW MOSI4, MOSI3, MOSI2, MOSI1
054755   MOSI4: EXIT
054756   %
054756   %       CHECK IF A COMPLETE BLOCK IS RECEIVED
054756   %
054756   MOSI3: A:=0
054757          IF T:=MOSW-4=0 THEN 5=:MOSW; A:=1 FI
054766          EXIT
054767   %
054767   %       INITIATE BLOCK RECEIVE (SEARCH FOR SYNC), CLEAR INPUT BUFFER
054767   %
054767   MOSI2: T:=HDEV; *EXR ST; EXR ST          % DUMMY READ
054772          2204; T:=HDEV+DCONT; *EXR ST
```

```
054776            2205; GO MOSI7
055000    %
055000    %       RECEIVER RESET, CLEAR BUFFER
055000    %
055000    MOSI1:  2204
055001    MOSI7:  T:=HDEV+DCONT;*EXR ST
055004            0=:MOSW
055005            A:=0 BONE 14; *MCL PID          % DISABLE DRIVER ON LEVEL 12
055010            GO CLBUF
055011    MOSI6:  EXIT
055012    %
055012    %       TIMER ROUTINE FOR MODEM INPUT
055012    %
055012    MOTMI:  4=:MOSW
055014            2204; T:=HDEV+DCONT; *EXR ST
055020            GO RTACT
055021    %
055021    %       IOTRANS ROUTINE FOR MODEM INPUT
055021    %
055021    MOTRI:  L+1
055022            IF BHOLD>0 THEN T:=MBSYMOD; GO TDBGET FI
055027            A:=-1; EXIT
055031    *,FILL
055037    %
055037    %       SET STATUS FOR MODEM OUTPUT :
055037    %
055037    % A=-1: CLEAR MODEM OUTPUT BUFFER
055037    % A= 0: START SENDING OF CURRENT BLOCK
055037    % A= 1: SET 377 - BYTES TO SEND BETWEEN DATA
055037    % A= 2: SET ASCII SYNC TO SEND
055037    % A= 3: SET EBCDIC SYNC TO SEND
055037    % A= 4: SYNCRONIZE ON 26 RECEIVED (ASCII)
055037    % A= 5: SYNCRONIZE ON 62 RECEIVED (EBCDIC)
055037    % A= 6: SYNCRONIZE ON 226 RECEIVED (ASCII)
055037    % A= 7: SET ASCII SYNC (226 TO SEND BETWEEN DATA)
055037    %
055037    MOSTO:  IF A+1<0 OR >10 THEN EXIT FI
055045            GOSW MOSCL,MOS0,MOS1,MOS2,MOS3,MOS4,MOS5,MOS6,MOS7
055057    MOS1:   777;  GO MOSI1
055061    MOS2:   426;  GO MOSI1
055063    MOS3:   462;  GO MOSI1
055065    MOS4:   1026; GO MOSI1
055067    MOS5:   1062; GO MOSI1
055071    MOS6:   1226; GO MOSI1
055073    MOS7:   626;
055074    MOSI1:  T:=HDEV-3;*EXR ST;EXIT          % UPDATE HARDWARE SYNCREGISTERS
055100
055100    MOSCL:  0=:MOWSTAT=:TMR=:DERROR
055103            GO CLBUF
055104
055104    MOSU:   5; T:=HDEV+DCONT; *EXR ST
055110            TTMR=:TMR; 2=:MOWSTAT; EXIT     % SET WAIT CONDITION
055115    %
055115    %       IOTRANS ROUTINE FOR MODEM OUTPUT
055115    %
055115    MOTRO:  IF T:=MOWSTAT-2=0 THEN EXIT FI  % CALLING PROGR WILL WAIT
055122            L+1; T:=MBSYMOD; GO TDBPUT
055125    %
055126    %       TIME-OUT SUBROUTINE FOR MODEM OUTPUT
055125    %
```

```
055125   MOTMO: 12=:DERROR
055127          "0"; T:=HDEV+DCONT; *EXR ST
055133          IF ISTATE>0 GO RTACT
055136          0=:MOWSTAT
055137          EXIT
055140   RBUS
055153   *"8HM01+8HM02+8HM03+8HM04+8HM05+8HM06
"055153
055153   %-------------------------------------------------------------------------------
055153   % 33.2        I O R E M
055153   %
055153   SUBR IOREM
055153   IOREM: IF X:=X.RTRES >< 0 THEN CALL RTENT; CALL FTIMQU FI
055157          CALL STUPR
055160   RBUS
055163
055163
055163   *"IDR4
```

```
"055163
055163   %=============================================================================
055163   % 33.11       P L O T T
055163   %
055163
055163
055163   %- - -  GRAPHIC I/O.
055163   % MONITOR CALL:              CALL PLOTT(X,Y,Z,DVN,FUNC)
055163   % APPL. LEVEL
055163
055163   SUBR PLOTT
055163
055163   DISP -6; INTEGER POINTER PLDRI; PSID
055163
055163   PLOTT: CALL GET5; IF D3/\177700=100 THEN 2; GO ERROR FI
055173          D3; CALL LOGPH
055175          IF X:=D=0 THEN 2;  GO ERORR FI      % TEST IF DEVICE PRESENT
055202          IF X.RTRES><CURPROG THEN 5; GO ERORR FI  % RESERVED BY THIS USER ?
055210          X."PLDRI"=:P                      % CALL RELEVANT INTERFACEROUTINE
055212   ERORR: A=:ZAREG;  GO RET                  % DEVICE NOT AVAILABLE
055214   RBUS
055221
055221
055221   * 8IXFR
"055221   %=============================================================================
055221   %
055221   % 33.24       R E A D C L O C K
055221   %
055221   %
055221   %    ROUTINE TO READ PANEL CLOCK ON N-100 AND UPDATE THE
055221   %    INTERNAL SINTRAN CLOCK.
055221   %
055221
055221   *"8PACL
"055221    @DEC
055221   INTEGER ARRAY PTBASE(0)
055221   DATA(0,0,1,1,1979); INTEGER ARRAY TBASE(0)       % TIME BASE FOR PANEL CLOCK
055226   INTEGER ARRAY DAYEAR:=(365,366,365,365)
055232   INTEGER ARRAY DAMONTH:=(31,28,31,30,31,30,31,31,30,31,30,31)
055246   @OCT
055246   INTEGER ADDYEAR,ADDMONTH,PERIOD
055251   INTEGER HDAYES,DAYES,HOURS,SECOND
055255   INTEGER HDAY2,SEC2
055257
055257   INTEGER PX9CL:=(X9CL1)
055260   INTEGER PX8CL:=(X9CL0,X9CL1)
055262   INTEGER X9CL0,X9CL1,X9CL2,X9CL3,X9CL4,X9CL5,X9CL6,X9CL7
055272   INTEGER ARRAY PXCLX(0)
055272   INTEGER PPXCL:=PX8CL
055273   INTEGER PPX3CL:=PX9CL
055274   INTEGER XCLNUN,XCLUN
055276   DOUBLE DXCLUN
055300   INTEGER ARRAY PLXCL:=(XCLNUN,XCLUN)
055302   *"
"055302    INTEGER PWRFREST              % POWERFAIL/RESTART FLAG
055303   INTEGER ERUCL:=0              % SET WHEN PANEL CLOCK INCORRECT
055304
055304   *"8PACL
"055304    SUBR READCLOCK
055304   INTEGER POINTER LREG
```

```
055305   READCLOCK:
055305           IF PWRFREST><0 THEN EXIT FI
055310           A:=L=:"LREG"
055312           *PIOF
055313           CALL 3RCLOCK; *PION              % IN POF AREA
055315           1=:PWRFREST; GO LREG
055320   RBUS
055322
055322   SUBR EX113M,EX112M
055322   EX113M: *PION; MON 113; PIOF; EXIT
055326   EX112M: *PION; MON 112; PIOF; EXIT
055332   RBUS
055332
055332   *"-8PACL
"055332
055332   %=============================================================================
055332   %
055332   % 33.25       S E T C L O C K
055332   %
055332   %  ROUTINE TO UPDATE PANEL CLOCK ON N-100.  THE CLOCK IS UPDATED
055332   %  WITH THE COMMAND @UPDAT OR THE SIMILAR MONITOR CALL (MON 111)
055332   %  AND THE COMMAND @CLADJ OR THE SIMILAR MONITOR CALL (MON 112)
055332   %
055332   % CALLED WITH MONITOR LEVEL DISABLED!!!!
055332   %
055332
055332   SUBR SETCLOCK
055332   INTEGER POINTER LREG
055333   SETCLOCK:
055333           IF PWRFREST><1 THEN EXIT FI
055340           A:=L=:"LREG"
055342           *POF
055343           CALL 3SETCLOCK
055344           *PON
055345           GO LREG
055346   RBUS
055350
055350
055350   %  PANEL PROCESSOR DRIVER
055350   SUBR PANEL
055350   INTEGER TIMOX:=-2000
055351   PANEL: *TRR 0
055352           X;=TIMOX; *TRA 0; BSKP ONE 140 DA; JNC *-2
055356           EXIT
055357   RBUS
055357   *"
"055357
055357   %=============================================================================
055357   %
055357   % 33.26       G E R D V
055357   %
055357   %  MONITOR CALL TO RETURN ERROR DEVICE (LOGICAL NUMBER) AND BY WHOM
055357   %  IT MIGHT BE RESERVED.
055357   %
055357
055357   SUBR GERDV
055357   INTEGER POINTER PTTNO:=TTNO
055360
055360   GERDV: CALL GETO; MLEV; *MST PIE
055363           T:=5ERRSEG; CALL MMEXY; A:=PTTNO; CALL MMEXY A=:ZAREG
```

```
055370          CALL LOGPH; A:=D.RTRES=:ZDREG; GO RET
055375    RBUS
055401
055401    @DEV 1
055401    @DEV (S-S-J)CDR2
055401
055401    %=============================================================================
055401    %       C D R 2
055401    %=============================================================================
055401
055401
055401
055401    *"8MT1+8MT2+8MT3+8MT4
"055401    %=============================================================================
055401    % 34.1        M G T M R    M T C L D
055401    %
055401    SUBR MGTMR,MTCLD
055401
055401
055401    % TIMEOUT SUBROUTINE FOR MAGTAPE
055401
055401    MGTMR: IF TRG/\77=10 OR =11 THEN EXIT FI %ADVANCE/RETURN TO EOF
055412           IF ><13 AND ><17 THEN 22=:TRG; "MFIN"=:"CLRG" FI
055424           IF A=17 THEN "MFIN"=:"DRIVER" FI
055431           GO MTMRSUB                          % STANDARD MASS STORAGE TIMEOUT
055432
055432    MTCLD: *PIOF
055433           HDEV+"164000+5"=:T; 20=:HSTAT; *EXR ST
055441    *"8MT1+8MT2+8MT3+8MT4 8DTMT
"055441             IF RTRES><0 THEN L=:D; CALL XRTACT; L:=D FI
055446           *PION; EXIT
055450    RBUS
055455
055455    *"99SM1+99SM2
"055455    %=============================================================================
055455    % 34.2        S T M R S U B
055455    %
055455    %       ROUTINE FOR SERVING STC-MAGTAPE
055455    %
055455    %       STMRSUB:                TIMEOUT-ROUTINE FOR REWINDING
055455    %                               WITH ASYNCHROUNOUS TIMEOUT FOR
055455    %                               EACH DRIVE EXECUTED BY INBT-DATAFIELDS
055455
055455    SUBR STMRSUB
055455
055455    STMRSUB:
055455           IF X := NRDYF = 0 THEN EXIT FI
055461           0=:ISTATE                           % SIMULATE RDATA
055462           X.STATUS BZERO 5WAIT =: X.STATUS
055465           EXIT                                % RETURN AFTER RESTARTING USER.
055466
055466    RBUS
055466
055466    %=============================================================================
055466    %       S M T B R E L
055466    %
055466    % MONITOR LEVEL ROUTINE TO RELEASE THE STC-DMA DATAFIELD AND DECREMENT
055466    % THE CALLING PROGRAM'S P-REGISTER
055466    %
055466    SUBR SMTBREL
```

```
055466   SMTBREL: X=:B:=X.RTRES; CALL BRELEASE
055471           IF X=CURPROG THEN
055474              *IRR ALEVB DP; AAA -1; IRW ALEVB DP
055477           ELSE
055500              X=:D:=X.RTDLGADDR; T:=0; *LDATX XTDPR
055504              A-1; *STATX XTDPR
055506              X:=D
055507           FI; "RETRA"=:MFUNC; GO MONEN
055512   RBUS
055516
055516   SUBR PT3MC144
055516   RBUS
055516
055516   *8MT1+8FDI1+8FDI2+8BFD1+8BFD2+8CAS1+8DMVC+8MT2+8DVE1+8DVE2+8DLP1+8DLP2+8CDLI+8HDMA+8MT3+8MT4 99MGT
"055516   %=============================================================================
055516   % 34.3          M C 1 4 4
055516   %
055516   %
055516   % CALL MAGTP(FUNC,CORAD,LOG.DEV,MAXWORDS,READWORDS)
055516   % ENTRY: B=DF-DATAFIELD
055516   %        PARAMETERS ARE IN DATAFIELD
055516   % EXIT: A=0 IF OK
055516   %
055516
055516   SUBR MC144
055516
055516   % VARIABLES IN I/O DATAFIELD:
055516
055516   DISP 25; INTEGER CASUN; PSID            % DEVICE UNIT NUMBER
055516
055516   % VARIABLES IN "DF" DATAFIELDS:
055516
055516   DISP 35
055516      INTEGER DMLOG                  % LOGICAL DEVICE NUMBER OF MASS STORAGE DEVICE
055516      INTEGER DMFLD                  % DATA FIELD ADDRESS OF MASS STORAGE DEVICE
055516      INTEGER IUNIT                  % DEVICE UNIT NUMBER
055516      INTEGER MDBUF                  % DEVICE BUFFER ADDRESS: LEAST SIG. 16 BITS
055516      INTEGER CMDBUF                 % DEVICE BUFFER ADDRESS: LEAST SIG. 16 BITS
055516                                     % WILL BE INCREMENTED WITH 1K FOR EACH K WORDS
055516                                     % TO READ/WRITE
055516      INTEGER MDFI                   % DEVICE BUFFER HEADER ADDRESS
055516      INTEGER FIMAX                  % NO. OF WORDS TO READ OR WRITE IF FLOPPY DISC
055516      INTEGER SVXERR=DMLOG           % TEMPORARLY SAVING OF X-REGISTER
055516      INTEGER TSAVE=MDBUF            % TEMPORARLY SAVING OF T-REGISTER
055516   PSID
055516
055516   %       DISPLACEMENTS IN INTERNAL BLOCK DEVICE DATAFIELDS.
055516
055516   DISP -3
055516      INTEGER DBLDN              % LOGICAL DEVICE NUMBER OF DBH RESERVED FOR THIS IBD. (ONLY FIRST).
055516      INTEGER INWORDS            % NO. OF WORDS IN BUFFER                                          INPUT
055516      INTEGER IDFOPP             % ADDRESS OF OUTPUT DATAFIELD
055516   PSID
055516   DISP 13
055516      INTEGER BREGC
055516   PSID
055516
055516   MC144: A:=L=:"MCLRG"
055520          CALL SPT3PIT; GO PT3MC144       % CONTINUE ON PIT3-SEGMENT
055522
```

```
055522    RBUS
055524
055524    * 8DMAS
"055524     *BLOCK=TEXIT; BUNLO=TEXIT
055524    * '
"055524
055524    $MAC
""055524
055524    * 8MT1+8CAS1+8DMVC+8CDLI+8FDI1+8FDI2+8BFD1+8BFD2+8MT2+8CAS2+8MT3+8MT4
"055524     %=============================================================================
055524    % 34.10       C B G E T   C B P U T   C A I C L   S M T R A   C A O C L
055524    %
055524    %
055524    % IOTRANS ROUTINE FOR MAG.TAPE, VERSATEC ON DMA, CASSETTE
055524    % AND FLOPPY DISC
055524    %
055524
055524    SUBR CBGET,CBPUT,CAICL,SMTRA,CAOCL
055524
055524    % VARIABLES IN I/O DATAFIELD
055524    DISP 24
055524    INTEGER LREGC                  % RETURN ADDRESS AFTER IOTRANS IS EXECUTED
055524    INTEGER CASUN                  % DEVICE UNIT NUMBER
055524    INTEGER CERROR                 % CURRENT ERROR CODE
055524    INTEGER LASTC                  % CURRENT CHARACTER
055524    INTEGER NOWRE                  % NUMBER OF CHARACTERS TO READ/WRITE
055524    INTEGER CPARM(4)               % PARAMETER LIST FOR MTRANS (FIRST WORD IN MEMORY...
055524                                   % BUFFER ADDRESS IS MBUFB.
055524    INTEGER MBUFB                  % BANK NUMBER IN WHICH THE DEVICE BUFFERS LIE.
055524    INTEGER MABUF                  % SECOND WORD IN MEMORY BUFFER ADDRESS
055524    DOUBLE MBPAG=MBUFB
055524    INTEGER VEFUNC                 % PRINT OR PLOT MODUS IF VERSATEC (10=PLOT,11=PRINT)
055524    INTEGER CIOLOG                 % LOGICAL DEVICE NUMBER OF I/O DATAFIELD
055524    INTEGER FDIADR                 % FLOPPY DISC ADDRESS
055524    INTEGER POINTER PLREGC=LREGC
055524    PSID
055524    DISP -1; INTEGER ADRBHEAD; PSID        % BUFFER HEADER ADDRESS
055524
055524    % VARIABLES IN DMA DEVICE DATAFIELD
055524    DISP 12; INTEGER MRTREF; PSID          % PROGRAM CALLING CLOSE
055524    DISP 13; INTEGER BREGC; PSID       % ADDRESS OF I/O DATAFIELD
055524    DISP 0; REAL RFMAX=MAX; DOUBLE DBHOLD=BHOLD; PSID
055524    INTEGER BREG=?
055524
055524    CBGET: A:=L=:LREGC; *PION              % LEVEL 4
055527           CALL CBF1
055530           IF BHOLD><0 THEN
055532              IF A=1 AND CERROR><0 THEN A=:DERROR FI
055540              *PIOF
055541              CALL DBGET; *PION
055543              MIN LREGC; GO PLREGC
055545           FI
055545           A:=B; *IRW MLEVB DB
055547           "CGMLEV"; *IRW MLEVB DP
055551           MLEV; *MST PID; WAIT
055554           CALL ERRFATAL
055555           *)FILL
055561
055561    % MONITOR LEVEL
055561
```

```
055561   CGMLEV:CALL CBF2; CALL MTRANS; CALL BRELEASE; CALL CBUNLOCK
055565          CALL MBFDISC; X:=BREGC;0=:X.CERRO
055570          IF TYPRING BIT 5FLOP THEN
055573             IF HSTAT NBIT 4 THEN
055576                IF A BIT 5 THEN A BONE 4=:HSTAT; 3=:CERRCODE; A:=0
055605                ELSE
055606                   X:=BREGC.CASUN; X:=FDIFORM(X); WDSCT(X)      % WORDS PER SECTOR ACCORDING TO FORMAT
055612                FI
055612             ELSE
055613                A:=0
055614             FI; A=:BREGC.NOWRE
055616          FI
055616          IF HSTAT BIT 4 THEN X:=:B; CALL MERRCODE; X:=:B
055624             IF T:=X.NOWRE=0 THEN A=:X.DERROR ELSE A=:X.CERROR FI
055632          FI
055632   CBGE1: X=:B;NOWRE SHZ 1=:BHOLD;0=:HENTE
055637          GO STUPR
055640
055640   ERR1:  A:=172                  % NO DEVICE BUFFER AVAILABLE
055641   ERR:   A=:DERROR; GO PLREGC
055643   *)FILL
055651
055651   INTEGER BREG
055652   INTEGER POINTER REGL
055653
055653   % RESERVE THE DFOPP-DATAFIELD AND RESERVE DEVICE BUFFER
055653
055653   CBF1:  T:=L=:"REGL"
055655          CALL CHDVBUF; GO CNBUF; GO L1
055660   CNBUF: CLOGDV; CALL LOGPH; IF A=0 THEN 2; GO ERR FI
055665          IF A.TYPRING BIT 5FLOP THEN
055671                X=:D; A:=X+CASUN+"FDIFORM"=:X; X.SO+D+"WDSCT"=:X
055702                X.SO SH 1=:MAX
055705          FI
055705          T:=CIOLOG; MAX SHZ -1; CALL G5BUF; GO ERR1
055712          AD=:MBPAG; A:=D=:BUFST; T=:ADRBHEAD                    % PUT ADDRESS OF DEVICE BUFFER AND DEVICE BUFFE
055716          0=:BHOLD=:FYLLE=:HENTE; MAX=:CFREE                     % IN THE FLOPPY INBYTE/OUTBYTE DATA FIELD.
055723   L1:    IF X:=DFOPP=0 GO REGL
055726          IF X.RTRES><RTREF THEN
055732             B=:D:=X; X:=T; CALL BRESERVE; D=:B
055737             IF A<0 THEN A:=205; GO ERR FI          % DEV. ALREADY RESERVED
055742          FI; GO REGL
055743
055743   CBF2:  A:=L=:"REGL"
055745          *IRR ALEVB DP
055746          A-1;*IRW ALEVB DP
055750          MAX SHZ -1=:NOWRE; CLOGDV; CALL LOGPH
055755          IF A=0 THEN 2; GO ERR FI        % NO  "DMA" DATAFIELD
055760          A:=:B;A=:BREG; CALL CBLOCK;X:=RTREF;CALL BRESERVE  % RESERVE "DMA" DATAFIELD
055765          IF A<0 THEN CALL FREXQU;CALL TOWQU; CALL CBUNLOCK
055771                GO STUPR FI               % WAIT FOR "DMA" DEVICE TO BE FREE
055772          IF TYPRING BIT 5FLOP THEN
055775             1=:BREG.NOWRE
056000             X:=X.CASUN; NFDIADR(X)=:BREG.FDIADR
056004          FI; BREG=:BREGC; A+"CPARM"; GO REGL
056010   *)FILL
056026
056026   CBPUT: X:=L=:LREGC;A=:LASTC; *PION
056032          CALL CBF1
056033          IF CFREE><0 THEN
```

```
056035              LASTC; *PIOF
056037              CALL DBPUT; *PION
056041              MIN LREGC; GO PLREGC
056043           FI
056043           A:=B; *IRW MLEVB DB
056045           "CBMLEV"; *IRW MLEVB DP
056047           MLEV; *MST PID; WAIT
056052
056052    % MONITOR LEVEL
056052    CBMLEV:CALL CBF2
056053           CALL MTRANS;CALL BRELEASE; CALL CBUNLOCK
056056           CALL MBFDISC; X:=BREGC
056060           IF HSTAT BIT 4 THEN X:=:B; CALL MERRCODE; X:=:B; A=:X.DERROR FI
056067           X=:B;MAX=:CFREE;0=:FYLLE=:BHOLD;GO STUPR
056075    *)FILL
056106
056106    % INCREMENT DISC ADDRESS IF FLOPPY DISC
056106    MBFDISC: IF TYPRING BIT 5FLOP THEN
056111                X:=BREGC.CASUN; MIN NFDIADR(X); 0/\0
056115             FI; EXIT
056116
056116
056116    % IOSET ROUTINE FOR MAG.TAPE, VERSATEC ON DMA, LINE-PRINTER, FLOPPY
056116    % AND CASSETTE
056116
056116
056116    INTEGER POINTER LREG=?
056116
056116    % ENTRY POINT FOR INPUT
056116    CAICL: IF A><-1 AND ><-2 THEN EXIT FI
056125    CAICF: A:=L=:"LREG"
056127           CALL CHDVBUF; GO CAIC1
056131           "CAIC4"; *IRW BLEVB DP
056133           CIOLOG; *IRW BLEVB DT
056135           BLEV; *MST PID
056137    CAIC1: 0=:BHOLD=:HENTE=:BUFST=:ADRBHEAD; MAX=:CFREE
056145           A:=0; GO LREG
056147    CAIC4: CALL R5BUF; CALL ERRFATAL; *WAIT
056152           CALL ERRFATAL
056153
056153    INTEGER SAVEB
056154    INTEGER REGB=?,XREG=?
056154    INTEGER POINTER LREG
056155
056155    % ENTRY POINT FOR OUTPUT
056155    CAOCL: IF A><-1 THEN IF A=-2 GO CAICF; EXIT FI
056164           IF BHOLD=0 GO CAICF
056166           A:=L=:"LREG"; CALL CHDVBUF; GO CAIC1
056172           X=:XREG
056173           A:=B=:REGB;CLOGDV;CALL LOGPH;A=:B
056200           X:=XREG; CALL CB2LOCK; X:=RTREF;CALL BRESERVE
056204           IF A<0 THEN            % WAIT FOR "DMA"DEVICE
056205                CALL FREXQU;CALL TOWQU; XREG=:B; ZPREG-1=:ZPREG
056214                CALL CBUNLOCK; GO RETSTUPR; *)FILL
056232           FI
056232           RTREF=:MRTREF; X:=REGB; X:=:B; CALL CHDVBUF; GO UT; X=:B
056241           IF TYPRING BIT 5FLOP THEN
056244              IF REGB.BHOLD><X.MAX THEN
056251                 A:=B=:SAVEB; X=:B
056254                 DO WHILE BHOLD><MAX
```

```
056260                  *PIOF
056261                  A:=0; CALL DBPUT; *PION
056264               OD; A:=SAVEB=:B
056267            FI
056267         FI; GO FILSK; *)FILL
056273
056273   INTEGER REGB,XREG
056275
056275   FILSK: REGB=:BREGC=:B; "SMTRA"=:MFUNC
056302         IF BHOLD BIT "0" THEN            % ODD NUMBER OF BYTES IN BUFFER
056305            *PIOF
056306            A:=0; CALL DBPUT; *PION
056311         FI
056311         IF TYPRING BIT 5MT THEN
056314            DO WHILE BHOLD<22             % RECORD OF AT LEAST 22 BYTES MUST
056320               *PIOF                      % BE WRITTEN ON MAG TAPE FOR IT TO
056321               A:=0; CALL DBPUT; *PION    % BE DISTINGUISHED FROM NOISE
056324            OD
056325         FI
056325         A:=BHOLD SHZ -1=:NOWRE; CALL RTACT
056331         X:=RTREF; CALL WDATA
056333   OUT:  XREG=:B; GO RET
056336   *)FILL
056344
056344   % SUBROUTINE TO SET UP A CALL TO THE MASS STORAGE DRIVER
056344   % MONITOR LEVEL
056344
056344   SMTRA: X=:B                            % LEVEL 3
056345         CLOGDV; CALL LOGPH; A=:X
056350         "SRETRA"=:X.MFUNC
056352         IF X.TYPRING BIT 5FLOP THEN
056355            X=:D; 1=:NOWRE; CASUN+X+"NFDIADR"=:X; X.S0; MIN X.S0; 0/\0
056367            A=:FDIADR; D=:X
056371         FI
056371         A:=B+"CPARM"; X=:B;CALL CMTRANS
056375         IF HSTAT BIT 4 THEN X:=:B; CALL MERRCODE; X:=:B ELSE A:=0 FI; A=:D
056406         IF CURPROG=MRTREF THEN
056412             A:=D; *IRW ALEVB DA
056414         ELSE
056415            A:=D; X:=MRTREF.RTDLGADDR; T:=0; *STATX XTDAR
056422         FI; X:=MRTREF
056423         "RETRANS"=:MFUNC; CALL BRELEASE; CALL CBUNLOCK; BREGC=:B
056431         "IORES"=:MFUNC
056433   OUT:  0=:BHOLD=:HENTE=:FYLLE; MAX=:CFREE
056440         CALL CHDVBUF; GO STUPR
056442         T:=CIOLOG; CALL R5BUF; CALL ERRFATAL; 0=:ADRBHEAD
056446         GO STUPR
056447
056447   RBUS
056466
056466
056466   *"8MT1+8MT2+8CAS1+8CAS2+8FDI1+8FDI2+8BFD1+8BFD2+8DLP1+8DLP2+8DMVC+8DVE1+8DVE2+8MT3+8MT4
"056466   %=============================================================================
056466   % 34.11        C M T R A N S    S R E T R A
056466   %
056466   %
056466   % SUBROUTINE TO START DRIVER ON LEVEL 11
056466   % MONITOR LEVEL
056466   % ENTRY: B=DATAFIELD ADDRESS
056466   %        A=ADDRESS OF PARAMETER LISTE
```

```
056466    %
056466    SUBR CMTRANS,SRETRA
056466
056466    DISP 12;INTEGER MRTREF;PSID  % RT PROGRAM CALLING CLOSE (IOSET)
056466
056466    CMTRANS: *IOF
056467             *IRW LV11B DX; TRR 10
056471             "STDRIV";*IRW LV11B DT
056473             A:=B; *IRW LV11B DB
056475             "SLV11"; *IRW LV11B DP
056477             A:=L=:"TRLREG"
056501             LV11;*MST PID
056503             X:=MRTREF;CALL WDATA
056505             *ION
056506             IF X=RTREF GO RWAIT; GO MONEN
056512
056512    % X=DATAFIELD ADDRESS
056512    SRETRA:  X=:B:=MRTREF;IF X=0 GO MONEN
056516             CALL RDATA;GO TRLREG
056520
056520    RBUS
056527
056527    * BHDMA
```

```
"056527   %
056527   %=============================================================================
056527   % 34.18       C L P U T   C L C L O S E
056527   %
056527
056527   %
056527   % IOTRANS ROUTINE FOR LINE-PRINTER AND VERSATEC
056527   %
056527
056527   SUBR CLPUT,CLCLOS
056527
056527   % VARIABLES IN I/O DATAFIELD
056527   DISP 24
056527   INTEGER LREGC                  % RETURN ADDRESS AFTER IOTRANS
056527   INTEGER CASUN                  % DEVICE UNIT NUMBER
056527   INTEGER CERROR                 % CURRENT ERROR NUMBER
056527   INTEGER LASTC                  % CURRENT CHARACTER
056527   INTEGER NOWRE                  % NUMBER OF WORDS TO TRANSFER
056527   INTEGER CPARM(4)               % PARAMETER LIST TO ABSTRANS MONITOR CALL.
056527   INTEGER MBUFB                  % BANK IN WHICH DEVICE BUFFERS LIE. FIRST WORD OF MEMORY ADDRESS FOR ABSTRANS
056527   INTEGER MABUF                  % DEVICE BUFFER ADDRESS RELATIVE TO START OF BANK. SECOND WORD OF BUFFER ADDR
056527   INTEGER VEFUNC                 % "PRINT/PLOT MODUS" FOR VERSATEC
056527   INTEGER CIOLOG                 % LOGICAL DEVICE NUMBER OF I/O DATAFIELD
056527   INTEGER MDATAF                 % ADDRESS OF MASS STORAGE DATAFIELD
056527   PSID
056527   DISP -1; INTEGER ADRBHEAD; PSID        % BUFFER HEADER ADDRESS
056527   % VARIABLE IN MASS STORAGE DATAFIELD
056527   DISP 12; INTEGER MRTREF; PSID          % PROGRAM CALLING CLOSE
056527   DISP 13; INTEGER BREGC; PSID  % ADDRESS OF I/O DATAFIELD
056527
056527   INTEGER POINTER REGL
056530
056530   CLPUT: *PION
056531          A=:LASTC:=L=:"REGL"
056534          CALL CHDVBUF; GO CNBUF; GO L1
056537   CNBUF: T:=CIOLOG; MAX SHZ -1; CALL G5BUF; GO ERR; A=:MBUFB; D=:A
056546          A=:BUFST=:MABUF; T=:ADRBHEAD
056551          0=:BHOLD=:FYLLE=:HENTE; MAX=:CFREE
056556   L1:    IF CFREE=0 THEN
056560                 A:=B; *IRW MLEVB DB
056562                 "CLMLEV"; *IRW MLEVB DP
056564                 MLEV; *MST PID; WAIT
056567   CLMLEV:       *IRR ALEVB DP
056570                 A-1; *IRW ALEVB DP
056572                 MDATAF=:B; X:=RTREF; CALL BRESERVE
056576                 IF A<0 THEN
056577                    CALL FREXQU; CALL TOWQU; GO STUPR
056602                 FI; BREGC=:B
056604                 MAX SHZ -1=:NOWRE; 0=:BHOLD=:FYLLE; MAX=:CFREE
056613                 A:=B+"CPARM";X:=MDATAF=:B; CALL MTRANS; CALL BRELEASE
056621                 BREGC=:B
056623                 GO STUPR
056624          FI
056624          *PIOF
056625          LASTC; CALL DBPUT; *PION
056630          MIN "REGL"; GO REGL
056632   ERR:   172=:DERROR; GO REGL  % NO DEVICE BUFFER AVAILABLE
056635
056635   *)FILL
056650   %
```

```
056650    % ROUTINE TO CLOSE LINE-PRINTER/VERSATEC
056650    %
056650
056650    INTEGER XREG,BREG
056652    CLCLOS: IF A><-1 THEN
056655                  IF A=-2 THEN
056660                     A:=L=:"REGL"; CALL CHDVBUF; GO L2
056664                     T:=CIOLOG; CALL R3BUFF; CALL ERRFATAL
056667    L2:              0=:ADRBHEAD; "REGL"=:L
056672                     0=;BHOLD=:FYLLE; MAX=:CFREE
056676                  FI; EXIT
056677            FI
056677            X=;XREG;=B=:BREG; MDATAF=:B; X:=RTREF; CALL BRESERVE
056706            IF A<0 THEN
056707                  CALL FREXQU; CALL TOWQU; XREG=:B
056713                  ZPREG-1=:ZPREG; GO RETSTUPR
056717            FI; X=:MRTREF; CALL WDATA; BREG=:B
056723            "CLTRA"=:MFUNC; CALL RTACT; XREG=:B; GO RET
056731    *)FILL
056745
056745    CLTRA: X=:B; "IORES"=:MFUNC
056750           CALL CHDVBUF; GO L4
056752           IF BHOLD><0 THEN
056754                 IF CFREE><0 AND PVEFUNC><10 THEN
056762                    *PIOF
056763                    12; CALL DBPUT; *PION
056766                    IF CFREE >< 0 THEN                                    --
056770                       *PIOF
056771                       215; CALL DBPUT; *PION
056774                    FI
056774                 FI
056774                 IF BHOLD BIT "0" THEN
056777                   *PIOF
057000                   A:=0; CALL DBPUT; *PION;
057003                   BHOLD
057004                 FI
057004                 A SHZ -1=:NOWRE; MDATAF=:B
057010                 "SRETRANS"=:MFUNC; BREGC+"CPARM"; CALL CMTRANS
057015                 "RETRANS"=:MFUNC; BREGC=:B
057021           ELSE X:=MDATAF=:B; X:=MRTREF; CALL RDATA; BREGC=:B
057030           FI
057030           T:=CIOLOG; CALL R5BUF; CALL ERRFATAL
057033    L3:    0=:BHOLD=;FYLLE=:ADRBHEAD; MAX=:CFREE
057040           IF CURPROG=MDATAF.MRTREF THEN
057045                 A:=0; *IRW ALEVB DA
057047           ELSE
057050                 X:=T.RTDLGADDR; T:=0; *STZTX XTDAR
057054           FI
057054           IF MDATAF.RTRES><0 THEN X=:B:=A; CALL BRELEASE FI
057062           GO STUPR
057063    L4:    MDATAF=:B; X:=MRTREF; CALL RDATA; BREGC=:B; GO L3
057072
057072    RBUS
057110
057110    %=============================================================================
057110    % 34.19       T R D L P
057110    %
057110    %
057110    % ENTRY POINT ON MONITOR LEVEL FOR LINE-PRINTER/VERSATEC DRIVER
057110    %
```

```
057110    SUBR TRDLP
057110    TRDLP: *POF
057111           CALL XTRDLP
057112    RBUS
057113
057113
057113    %-=========================================================================
057113    % 34.20       D M L P C    D L P T M R
057113    %
057113    % CLEAR-DEVICE AND TIME-OUT SUBROUTINE FOR LINE-PRINTER AND VERSATEC
057113    %
057113
057113    SUBR DMLPC,DLPTMR
057113
057113    % VARIABLES IN MASS STORAGE DATAFIELD
057113    DISP -41
057113    INTEGER VEFLG                  % FLAG=1 IF VERSATEC ELSE EQUALS ZERO
057113    INTEGER VEMOD                  % FUNCTION CODE IF NOT WRITE ON VERSATEC
057113    INTEGER POINTER CHCONV         % ADDRESS OF CHARACTER CONVERTING ROUTINE
057113    PSID
057113                                           % IN I/O DATAFIELD
057113    DISP -24
057113    INTEGER CXMAX                  % MAXIMUM OF BYTES IN EXTRA-BUFFER
057113    INTEGER CXBHOLD                % ACTUAL NUMBER OF BYTES IN EXTRA-BUFFER
057113    INTEGER CXBUFST                % ADDRESS OF EXTRA-BUFFER
057113    INTEGER CXHENTE(2)             % HENTE-POINTER IN EXTRA-BUFFER
057113    INTEGER CIOXC                  % CODE TO SET OUT IN IOX-WRITE-CONTROL
057113    INTEGER IOXWC                  % CODE FOR IOX-WRITE-CONTROL
057113    INTEGER IOXRS                  % CODE FOR IOX-READ-STATUS
057113    INTEGER IOXWD(2)               % CODE FOR IOX-WRITE-DATA
057113    INTEGER LP5MF                  % ADDRESS OF MONITOR FUNCTION ROUTINE
057113    INTEGER CBHOLD                 % CURRENT NUMBER OF BYTES IN DEVICE BUFFER
057113    INTEGER CHENTE                 % HENTE-POINTER IN DEVICE BUFFER
057113    INTEGER CBUFST                 % DEVICE BUFFER ADDRESS
057113    PSID
057113
057113    DMLPC: *IOF
057114           T:=HDEV+"DCONT"; 20; *EXR ST
057120           0=:CXBHOLD=:CXHENTE=:CBHOLD=:CHENTE
057124           IF RTRES><0 THEN L=:D; CALL XRTACT; L:=D; FI
057131           *ION; EXIT
057133
057133    % TIMER SUBROUTINE FOR LINE PRINTER/VERSATEC
057133    DLPTMR: TTMR=:TMR
057135           IF VEFLG><0 THEN
057137              IF PVEFUNC=10 THEN 101; GO L1 FI
057145           FI; A:=1
057146    L1:    T:=IOXWC; *EXR ST; EXIT
057151    RBUS
057152    *"
"057152    %=========================================================================
057152    % 34.21       C H D V B U F
057152    %
057152    % SUBROUTINE TO CHECK IF A DEVICE BUFFER IS RESERVED FOR THIS LOGICAL
057152    % DEVICE UNIT BY THE PROGRAM WHICH HAS RESERVED THE LOGICAL
057152    % DEVICE UNIT
057152    %
057152    % ENTRY:       B=ADDRESS OF I/O DATAFILELD
057152    % EXIT:        NO DEVICE BUFFER RESERVED
057152    % EXIT A:      DEVICE BUFFER RESERVED
```

```
057152   %
057152   SUBR CHDVBUF
057152   DISP -1; INTEGER ADRBHEAD; PSID          % ADDRESS OF BUFFER HEADER
057152   DISP 5; INTEGER LGIOLOG; PSID            % LOGICAL DEVICE NUMBER IN BUFFER
057152   DISP 40; INTEGER CIOLOG; PSID            % LOGICAL DEVICE NUMBER IN DATAFIELD
057152
057152   CHDVBUF:
057152           X=:D
057153           IF ADRBHEAD=0 OR A.RTRES><RTRES OR X.LGIOLOG><CIOLOG THEN
057166              X:=D
057167              EXIT
057170           FI
057170           X:=D; EXITA
057172   RBUS
057172
```

```
057172
057172
057172   %=============================================================================
057172   % 34.22      P S T W O R D    P L D W O R D    D B S T W    D B L D W
057172   %
057172   % SUBROUTINES TO READ/WRITE VALUE IN ANY LOCATION IN EXISTING MEMORY
057172   %
057172   %       ROUTINES DBSTW AND DBLDW RETURN WITH PAGING-OFF AND INTERRUPT UNCHANGED
057172   %       - CALLED FROM ROUTINES OPERATING ON DEVICE BUFFERS.
057172   %
057172   %       PSTWORD AND PLDWORD RETURN WITH ION AND PAGING-OFF
057172   %       - CALLED FROM DMA-DRIVERS WITH ION.
057172   %
057172   % ENTRY: T=VALUE TO STORE IF PSTWORD OR DBSTWORD.
057172   %         AD=ADDRESS OF LOCATION TO ACCESS.
057172   %
057172   % EXIT:  T=VALUE IF PLDWORD OR DBLDWORD.
057172   %         AD IS INCREMENTED BY ONE
057172   %
057172
057172   SUBR PSTWORD,PLDWORD,DBSTWORD,DBLDWORD
057172
057172   INTEGER SVT,SVA,SVD,SVX,FLGX,PITAB; DOUBLE SVAD=SVA; REAL SVTAD=SVT
057200   INTEGER 5BFAD(0); *5BFPA@12
057201   INTEGER 5BFXX(0); *177400+5BFPA
057202
057202   DBSTWORD: K:="0"; GO PS1
057204   DBLDWORD: K:="0"; GO PL1
057206   PSTWORD: *IOF
057207            K:=1
057210   PS1:     0=:FLGX; GO FELLS
057212   PLDWORD: *IOF
057213            K:=1
057214   PL1:     T:=1=:FLGX
057216   FELLS:   X=:SVX
057217               IF X:=FLGX=0 THEN
057222                  *DEPO
057223               ELSE
057224                  *EXAM
057225               FI; *POF
057226            D+1; A:=A+C; X:=SVX
057231            IF K THEN; *ION        % PSTWORD/PLDWORD
057234                 FI; EXIT
057235   RBUS
057235   *"8FDI1+8FDI2
"057235    %=============================================================================
057235   % 34.24      I M T R I    I M T R O
057235   %
057235   %
057235   %       MTRANS ROUTINES FOR BLOCK ORIENTED INTERNAL DEVICES. ONLY FUNCTION CODES
057235   %       0, 1 AND 21 ARE VALID.
057235   %
057235   SUBR IMTRI,IMTRO
057235
057235   DISP -3
057235   INTEGER DBLDN                  % DBH LOGICAL DEVICE NUMBER: USED AS A FLAG AND FOR LOCKING / UNLOCKING.
057235   INTEGER INWORDS                % NO. OF WORDS IN BUFFER
057235   INTEGER IDFOPP                 % ADDRESS OF OUTPUT DATAFIELD
057235   INTEGER POINTER CIFUNC         % POINTER TO FUNCTION CODE
057235   DOUBLE POINTER PDCADDR(2)      % POINTER TO MEMORY (BUFFER) ADDRESS
```

```
057235    INTEGER POINTER PNWORDS      % POINTER TO THE PAR. NO. OF WORDS
057235    PSID
057235    DISP 13
057235
057235    INTEGER BREGC
057235
057235    PSID
057235
057235    INTEGER POINTER LREG; INTEGER XREG
057237
057237    IMTRI: A=:X; X:=:B; A:=L=:"LREG"; 0=:X.HSTAT
057244           IF CIFUNC /\ 77 = 0 THEN
057247    %
057247    %          READ FROM INTERNAL DEVICE.
057247    %
057247               IF X.BREGC >< 0 THEN
057251                  0=:X.BREGC                          % BREGC IS USED AS A FLAG: IF >< 0 THERE IS DATA WAITING
057252                  IF X.INWORDS < PNWORDS THEN         % TO BE READ IN THE IBD. IF = 0 THERE IS NONE.
057256                     A=:PNWORDS                       % NUMBER OF WORDS IN IBD < NUMBER SPECIFIED.
057257                  FI
057257                  X=:XREG
057260                  IF X.IDFOPP.RTRES >< 0 THEN
057263                     X=:B:=A; CALL RDATA              % OUTPUT DATAFIELD IS RESERVED.
057266                     A:=XREG=:B; A:=DBLDN; CALL LOGPH
057272                     IF A = 0 THEN CALL ERRFATAL FI
057274                     A=:B; 1=:DKFUN                   % CHANGE FUNCTION CODE IN DBH TO WRITE
057277                     CALL BRESERVE                    % RESERVE DBH FOR RT-PROG OWNING OUTPUT DATFIELD.
057300                     IF A < 0 THEN
057301                        CALL FREXQ; CALL TOWQU        % PUT THE RT-PROG INTO WAITING QUEUE FOR DBH.
057303                     FI
057303                  FI
057303                  XREG=:B; X:=RTRES; GO LREG
057307               FI
057307               X=:B; A:=DBLDN
057311    CALLWDATA:CALL LOGPH
057312               IF A = 0 THEN CALL ERRFATAL FI
057314               B=:D; A=:B; X:=RTRES; CALL BRELEASE
057320               B:=D; X:=RTRES; CALL WDATA           % SET THE CALLING PROGRAM INTO I/O WAIT.
057323               IF X=RTREF THEN
057326                  *IRR ALEVB DP
057327                  A-1; *IRW ALEVB DP                % DECREMENT THE P-REG ON APPLICATION LEVEL.
057331                  GO RWAIT
057332               FI
057332               X:=X.RTDLGADDR; T:=0; *LDATX XTDPR
057335               A-1; *STATX XTDPR
057337               GO MONEN
057340            FI
057340    %
057340    %          NOT READ: IS FUNCTION CLEAR DEVICE?
057340    %
057340            IF A=21 THEN
057343               0=:X.DBLDN=:BREGC                    % CLEAR DEVICE: CLEAR DBH LDN AND DATA IN BUFFER FLAG.
057345            ELSE
057346               20=:X.HSTAT                          % INDICATE ILLEGAL FUNCTION.
057350            FI
057350            X=:B:=RTRES; EXIT
057353    %
057353    %          WRITE TO IBD.
057353    %
057353    IMTRO: A=:X; X:=:B; A:=L=:"LREG"; 0=:X.HSTAT
```

```
057360          IF CIFUNC/\77 >< 1 THEN
057365             20=:X.HSTAT; X=:B:=RTRES; EXIT          % FUNCTION NOT WRITE: INDICATE ILLEGAL FUNCTION CODE.
057372          FI
057372          X=:XREG
057373          IF X.IDFOPP.BREGC >< 0 THEN
057376             X:=XREG; A=:B; A:=X.DBLDN
057401             GO CALLWDATA                            % WAIT UNTIL DATA HAS BEEN READ BEFORE WRITING NEW DATA.
057402          FI                                         % X-REG NOW POINTS TO INPUT PART.
057402          1=:X.BREGC                                 % FLAG IBD AS CONTAINING DATA.
057404          PNWORDS=:X.INWORDS
057406          IF X.RTRES >< 0 THEN
057410             X=:B; A=:X; CALL RDATA; A:=DBLDN; CALL LOGPH
057415             IF A = 0 THEN CALL ERRFATAL FI
057417             A=:B; 0=:DKFUN; CALL BRESE              % FUNCTION TO BE CONTINUED IS READ.
057422             IF A < 0 THEN
057423                CALL FREXQ; CALL TOWQU
057425             FI
057425          FI
057425          X:=XREG=:B:=RTRES; GO LREG
057431    RBUS
057445
057445    *"
"057445
057445
```

```
057445    %
057445
057445    *"
"057445
057445
057445    * MERRC
"057445    %=============================================================================
057445    % 34.25       M E R R C O D E
057445    %
057445    %
057445    % SUBROUTINE TO DECODE HARDWARE STATUS TO ERROR CODE
057445    % ANY LEVEL
057445    %
057445    % ENTRY: A=HARDWARE STATUS, X=ADDRESS OF DATAFIELD
057445    % RETURN: A=ERROR CODE, D IS DESTROYED
057445    %
057445    SUBR MERRCODE
057445    DISP 22; INTEGER DIS22; PSID
057445    INTEGER CINSTR(0); *COPY SA DB
057446    MERRCODE; A=:D; IF X.CERRCODE><0 THEN 0=:X.CERRCODE; EXIT FI
057453            IF D=-1 THEN A:=174; EXIT FI          % ILLEGAL PARAMETER IN CALL
057460            IF X.TYPRING BIT 5IBDV THEN A:=D; EXIT FI
057465            IF A BIT 5FLOP THEN
057467                  IF X.DIS22-CINSTR><0 THEN 171; EXIT FI  % NEW FLOPPY (BIG-FLOPPY)
057474                  IF D BIT 10 THEN 204; EXIT FI
057500                  IF D BIT 11 THEN 175; EXIT FI
057504            FI                                                                  --
057504            IF A NBIT 5MT THEN A:=171; EXIT FI % MASS.DEV.ERROR
057510
057510    % MAG. TAPE
057510            IF X."TRNSF"><"HMAGT" THEN       % IF "SMAGT" OR "TMAGT" !!
057514                  IF D NBIT "0" THEN 204; EXIT FI      % NOT READY
057520            ELSE
057521                  IF D NBIT 16 THEN 204; EXIT FI       % NOT READY
057525            FI
057525            IF D BIT 7 THEN A:=3; EXIT FI       % END OF FILE
057531            IF D BIT 11 THEN A:=146; EXIT FI              % END OF TAPE
057535            IF D BIT 14 THEN A:=163; EXIT FI            % OVERFLOW IN READ
057541            IF D BIT 13 THEN A:=164; EXIT FI            % DMA ERROR
057545            IF X."TRNSF"><"HMAGT" THEN          % NOT HEWLET-PACKARD.
057551               IF D BIT 6 THEN A:=165; EXIT FI     % BAD DATABLOCK
057555               IF D BIT 5 THEN A:=166; EXIT FI          % CONTROL/MODUS WORD ERROR
057561            ELSE                                % H.P. MAG.TAPE
057562               IF D BIT 12 THEN A:=167; EXIT FI    % PARITY ERROR
057566               IF D BIT 6 THEN A:=170; EXIT FI          % LRC ERROR
057572            FI; A:=171; EXIT                            % MASS STORAGE DEVICE ERROR
057574    RBUS
057576    *"
"057576
057576
057576
057576
057576
057576    %=============================================================================
057576    % 34.26       M 8 O U T B   M O U R E T
057576    %
057576    % MONITOR CALL TO OUTPUT 8 BYTES
057576    % INBT/OUTBT LEVEL
057576    SUBR BBINSYN,B8OUSYN,BM8OUT,BB8OUT
057576    RBUS
```

```
057576   %
057576   SUBR M8OUTB,MOURET,M8OTERM,CH8BY,M8OINDV,B8OUT,B8OTERM,B8OINDV,CH4WO,T2P01,T2P02
057576   INTEGER POINTER PTTNO:=TTNO,PBCHFLAG:=BCHFLAG
057600   M8OUTB: B:=0; GO FELLS
057602   B8OUT: B:=1
057603   FELLS: *IRR ALEVB DT
057604          IF X:=BACKGROUND><0 AND X:=PBCHFLAG=0 THEN
057611                IF A=1 THEN A:=PTTNO FI
057615          FI; IF A=0 GO MOURET; IF A=:D/\177700=100 GO E240  % NOT ALLOWED ON FILE
057624          A:=D; CALL LOGPH; IF D=0 GO E240
057630          D:=:B                                          % D IS "OLD CFLAG"
057631          IF X:=RTREF><RTRES THEN 5; GO ERR FI
057637   @LIB CXCPU
057637          TYPRING; *PIOF
057641   @ELIB
057641   @LIB CXCPU-,
057641          IF A BIT 5TERM THEN
057643                IF D=0 GO M8OTERM;  GO B8OTERM
057646          FI
057646          IF A BIT 6BAD THEN
057650                IF D=0 THEN
057652   T2P01:          GO BM8OUT; 0/\0
057654                ELSE
057655   T2P02:          GO BB8OUT; 0/\0
057657                FI
057657          FI
057657   *"8CLI1+8CLI2+8CLI3+8CLI4+8CLI5+8CLI6
"057657           *IRR ALEVB DT
057660          IF A>177 AND A<300 THEN
057666                IF D=0 GO M8OINDV; GO B8OINDV
057671          FI
057671   *"8VIPS
"057671    E240:  240
057672   ERR:   *IRW ALEVB DA
057673   L1:    *PION; WAIT; JMP * 1; IOF; WAIT
057700   MOURET: *IRR ALEVB DP; AAA 1; IRW ALEVB DP
057703          GO L1
057704   RBUS
057721
057721   @ICR
057721   SUBR M8INTERM,B8INTERM,B8IINDV,M8IINDV,M8INCOM,B4INTERM,
057721        B4IINDV,B4WICOM,M8IBTERM,M8IBCOM,BB4INW,BB8INP;
057721   RBUS
057721   @CR;
057721   SUBR M8INB,B8INB,B4INW,T8INP,T2P03,T2P04,T2P05
057721   INTEGER POINTER PTTNO:=TTNO,PBCHFLAG:=BCHFLAG
057723   M8INB: B:=0; GO FELLS
057725   B8INB: B:=1; GO FELLS
057727   B4INW: B:=2; GO FELLS
057731   T8INP: B:=3
057732   FELLS: *IRR ALEVB DT
057733          IF X:=BACKGROUND><0 AND X:=PBCHFLAG=0 THEN
057740                IF A=1 THEN PTTNO FI
057744          FI; IF A=0 GO MOURET
057746          IF A=:D/\177700=100 GO E240                % NOT ALLOWED ON FILE
057753          A:=D; CALL LOGPH; IF A=0 GO E240
057756          B=:D:=A                                                % D IS "OLD CFLAG"
057760          IF X:=RTREF><RTRES THEN 5; GO ERR FI; *PIOF
057767          IF TYPRING BIT 5TERM THEN
057772                D GOSW M8INTERM,B8INTERM,B4INTERM,M8IBTERM
```

```
057777            FI
057777            IF A BIT 5BAD THEN
060001                  IF D=0 GO M8INTERM
060003                  IF D=1 THEN
060006    T2P03:           GO M8INTERM; 0/\0
060010                  FI
060010                  IF D=2 THEN
060013    T2P04:           GO BB4INW; 0/\0
060015                  FI
060015                  IF D=3 THEN
060020    T2P05:           GO BB8INP; 0/\0
060022                  FI
060022            FI
060022            IF A BIT 5COM THEN
060024                  D GOSW M8INCOM,M8INCOM,B4WICOM,M8IBCOM
060031            FI
060031            *IRR ALEVB DT
060032            IF A>177 AND A<300 THEN
060040                  D GOSW M8IINDV,B8IINDV,B4IINDV
060044            FI
060044    *"8VIPS
"060044     E240:  240
060045    ERR:    *IRW ALEVB DA; PION; WAIT
060050            CALL ERRFATAL
060051    RBUS
060075
060075
060075    %-------------------------------------------------------------------------------
060075    %
060075    %
060075    SUBR GRTDA,GSGNO,GRTNA,3GRTDA,3GSGNO,3GRTNA
060075
060075    GSGNO: T:="P0"; CALL GET0; CALL S3NOALT; GO 3GSGNO
060101    GRTDA: T:="P0"; CALL GET0; CALL S3NOALT; GO 3GRTDA
060105    GRTNA: CALL GET1; CALL S3NOALT; GO 3GRTNA
060110    RBUS
060116
060116
060116    %-------------------------------------------------------------------------------
060116    % 34.29       I P R I V
060116    %
060116    SUBR IPRIV
060116    INTEGER OBREG,NBREG,CAREG
060121    IPRIV: CALL GET0
060122           T:=ZTREG; X:=ZXREG; A:=ZDREG=:D
060126           A:=B=:OBREG:=ZAREG=:CAREG:=ZBREG=:B:=CAREG
060135           *EXR ST
060136           A=:CAREG:=B=:NBREG:=OBREG=:B
060143           A:=NBREG=:ZBREG:=D=:ZDREG
060147           T=:ZTREG; X=:ZXREG; CAREG=:ZAREG
060153           GO RET
060154
060154    RBUS
060156
060156    *"8CXHD+8C1X2+8C2X2+8C3X2+8C4X2
```

```
060156
060156  %-----------------------------------------------------------------------
060156  % 34.31     X T L X    S T U S P   S T L 1 2   S T L 1 3
060156  %
060156  %       MONITOR CALL HDLC
060156  %
060156  %-----------------------------------------------------------------------
060156  %
060156  %
060156  %PUROSE: SIMULATE XTLX MONITOR CALLS FOR COMMUNICATION
060156  %         WITH HDCL-DRIVER.
060156  %
060156  %ARGUMENTS:
060156  %
060156  %ENTRY: X-DEMFIELD, B-PARAMPOINTER
060156  %       PT=0, APT=USER PT, LEV 3 ENABLED, PON, ION
060156  %
060156  %EXIT: WHEN ENTRING THE CODE IN POF (HDLC),THE B-REG WILL
060156  %       POINT TO THE APPROPRIATE BASEFIELD,WHICH IN TURN
060156  %       WILL CONTAIN THE USER CALL PARAMETERS.
060156  %
060156  %
060156  %CALLING SEQUENCE:
060156  %
060156  %       A-REG - POINTER TO PARAM ADDRESSES
060156  %       MON  HDLC
060156  %       POINTER TO PARAM1
060156  %       POINTER TO PARAM2
060156  %       POINTER TO BUFFER
060156  %       POINTER TO PARAM4
060156  %       POINTER TO PARAM5
060156  %
060156  %       P1 - FUNCTION. SEND OR RECEIVE
060156  %       P2 - PORT NO. (LOGICAL UNIT NO.)
060156  %       P3 - BUFFER
060156  %       P4 - BYTECOUNT
060156  %       P5 - P1 - 1  MAX BYTECOUNT OF MESSAGE
060156  %            P1 - 2  WAIT/CONTINUE IF NO MESSAGE IN QUEUE
060156  %
060156  %
060156  DISP -9
060156          INTEGER HXDOK                   %LOCK,OPERATED BY X21(1-LOCKED,0-OPEN)
060156          INTEGER HXTMO                   %ADDRESS OF TIMEOUT-ROUTINE IN POF
060156  PSID
060156  SUBR XTLX,STUSP,IMHDLC
060156  %
060156  %       TEMPORARY WORKING AREA
060156  %
060156  INTEGER ARRAY POINTER TDAFI                     %CURRENT DATAFIELD
060157  INTEGER ARRAY POINTER TDEFI                     %DEMANDFIELD
060160  INTEGER TEMP
060161  INTEGER BFADR
060162  %
060162  XTLX:   *BSET ZRO
060163          CALL GAPIT; CALL DALTON
060165          A:="P2"=:BFADR                          %USER BUFFER ADDRESS
060167          IF P0 = FRECV OR A=FMXRECV THEN
060176                A:="P3"=:TEMP                     %RETURN ADDRESS OF BYTECOUNT
060200                CALL GET5; *BSET ZRO
060202                TEMP=:X.D3
```

```
060204          ELSE
060205                IF PO = FSEND THEN
060211                   CALL GET5
060212                ELSE
060213                   T:=EFUNC; GO ERET    %ILLEGAL FUNCTION
060215                FI
060215          FI; *BSET ZRO
060216          BFADR=:X.D2
060220          X=:"TDEFI"
060221    %
060221    %     LEV 3 DISABLED, TURN OFF PAGING
060221    %
060221          CALL ALTON; *POF                          %SET ALT PAG AND KEEP IT
060223    %
060223    %     FIND DATAFIELD AND RESERVE IT
060223    %
060223          A:=X.D1; CALL LOGPH;
060225          IF A = 0 THEN A:=D FI                     %TRY OUTPUT-FIELD
060227          IF A = 0 THEN T:=E2; GO ERET; FI
060232          A=:"TDAFI"=:B
060234          IF X.TYPRING >< 0 THEN T:=EDEVN; GO ERET FI          %*81F*
060240          IF HXCC >< HXCOD THEN T:=EDEVN; GO ERET FI  % CORRECT DATAFIELD ?
060246          IF CURPR><RTRES THEN                      % *80B*
060252            T:=E1;  GO ERET;                        % *80B* *81F*
060254          FI                                        % *80B*
060254    %
060254    %     COPY DATA TO DATAFIELD AND RELEASE DEMFIELD
060254    %
060254          A:="TDAFI"+OFSET=:"TDAFI"
060257          FOR X:= ST TO SP DO TDEFI(X)=:TDAFI(X) OD;
060267          A:="TDEFI"=:B
060271          X:=CURPROG; CALL BRELEASE;
060273          A:="TDAFI"-OFSET=:"TDAFI"=:B              %DATAFIELDPOINTER
060277    %
060277    %
060277          CALL HDLC                                 % GO TO POF AREA
060300    %
060300    %
060300    %     RETURN SEQUENCE
060300    %
060300    %     ARGUMENTS:
060300    %           B - DATAFIELDPOINTER
060300    %           T = 0, OK, GIVE SKIP RETURN
060300    %           MESSID = MESSAGE IDENTIFICATOR
060300    %
060300    %     T ><0, ERROR, NO SKIP RETURN
060300    %
060300    %     POF
060300    %
060300    HBACK: IF T><0 THEN
060302               B+OFSET
060303    ERET1:     T=:ZAREG; GO RET                     %*81F*
060305          FI
060305          A:=MESSID; B+OFSET; A=:ZAREG
060310          ZPREG+1=:ZPREG; GO RET
060314    ERET: X=:B; GO ERET1                            % *81F*
060316    %
060316    %     ROUTINE FOR STORING IN USER PARAM
060316    %
060316    %     T - VALUE TO BE STORED
```

```
060316   %
060316   %       USER ADDR. IN DDD3 IN DATAFIELD
060316   %
060316   DISP 0; INTEGER HNULL; PSID
060316   STUSP: MLEV; *IOF; MST PIE; PION                    %MUST ALLOW PAGEFAULT
060322          X:=B+OFSET; *BSET ZRO
060325          CALL GAPIT; CALL DALTON; *BSET ZRO
060330          A:=DDD3; A=:B; *BSET ONE
060333          T=:HNULL; X=:B; *BSET ZRO                    %STORE IN USER PARAM
060336          CALL ALTON; B-OFSET
060340          MLEV; *MCL PIE; POF
060343          GO IMHDLC                                    % CONTINUE IN ROUTINE HDLC IN POF
060344   RBUS
060357
060357   %       START LEVEL 12/13 IN POF AREA AT ADDRESS IN T-REG
060357   %       B-REG FROM CALLIN LEVEL IS COPIED
060357   %       RETURN TO CALLER WITH PIOF
060357
060357   SUBR STL12                                          %*80B*
060357   STL12: "SLV12"; *IOF; IRW LV12B DP                  %*80B*
060362          A:=B; *IRW LV12B DB                          %*80B*
060364          A:=T; *IRW LV12B DT                          %*80B*
060366          LV12; *MST PID; PION; PIOF; EXIT             %*80E*
060373   RBUS                                                %*80B*
060375   SUBR STL13                                          %*80B*
060375   STL13: "SLV13"; *IOF; IRW LV13B DP                  %*80B*
060400          A:=B; *IRW LV13B DB                          %*80B*
060402          A:=T; *IRW LV13B DT                          %*80B*
060404          LV13; *MST PID; PION; PIOF; EXIT             %*80B*
060411   RBUS                                                %*80B*
060413
060413
060413   %=========================================================================
060413   % 34.32      H D T M 2       H D T M 3
060413   %       OUTPUT TIMEOUT ROUTINE
060413   %
060413   %PURPOSE:     ACTIVATE TIMEOUT ROUTINE ON LEVEL 12 IN POF AREA
060413   %
060413   SUBR    HDTM2
060413   HDTM2: T:=HXTMO; GO STL12                           % *80B*
060415   RBUS
060416   %
060416   %
060416   %        INPUT TIMEOUT ROUTINE
060416   %
060416   %        PURPOSE: ACTIVATE TIMEOUT ROUTINE ON LEVEL 13
060416   %
060416   SUBR X21IN,X2STA,X21TO
060416   RBUS
060416   SUBR HDTM3
060416   HDTM3: T:=HXTMO; GO STL13                           % *80B*
060420   RBUS
060421   %
060421   %       SOME ROUTINES TO GET INTO PIT3 FROM POF
060421
060421   SUBR X42ST,X321T,D3CHA,S3L13,D3CHA,X321I,X3T12,X3T13
060421   INTEGER SAVL,SAVL1,SAVL2
060424   INTEGER LOGBA,PHBAS                    % LOGICAL AND PHYSICAL BASE
060426   FFF:    X:="SG41"; A:=X.LOGADR/\77=:LOGBA
060432           X:=X.BPAGLINK; T:=CORMBANK; *LDATX DPAGP
```

```
060435            A=:PHBAS; EXIT
060437    X32ST:  *PON
060440            GO X2STA
060441    X321T:  *PON
060442            GO X21TO
060443    X321I:  *PON
060444            GO X21IN
060445    O3CHA:  A:=L=:SAVL; X=:D; CALL FFF; A:=0; AD SHZ 6; A-LOGBA+PHBAS
060455            AD SHZ -6; D=:X; *POF
060460            CALL OCHAIN; *PON
060462            A:=SAVL=:P
060464    S3L13:  A:=L=:SAVL1; *POF
060467            CALL STL13; *PON
060471            A:=SAVL1=:P
060473    D3CHA:  A:=L=:SAVL2; *POF
060476            CALL DICHAIN; GO NSKIP
060500            MIN SAVL2; P+0
060502            X=:D; CALL FFF; A:=0; AD SHZ 6; A-PHBAS+LOGBA; AD SHZ -6; D=:X
060512    NSKIP:  *PON
060513            A:=SAVL2=:P
060515    X3T12:  *POF
060516            GO DRIVER
060517    X3T13:  *POF
060520            T=:P
060521    RBUS
060532
060532
060532    %=============================================================================
060532    %       I N B X 2 1
060532    %
060532    % SUBROTINE TO SET UP THE PHYSICAL ADDRESS OF THE "X21-BUFFERS" INTO
060532    % THE "X21" DATAFIELDS
060532    %
060532    % CALLED FROM OLDSTART
060532    %
060532    SUBR INBX21
060532
060532    INTEGER ARRAY ADRX21(0)
060532    *"8C1X2
"060532     *       X21F1;X2S01;X2E01
060535    *"8C2X2
"060535     INTEGER ARRAY EAX21(0)
060535    *"8C1X2+8C2X2+8C3X2+8C4X2+8C5X2+8C6X2
"060535     INTEGER CINDX,CADR,16BADR; DOUBLE DCAD=CADR
060540    DISP 0; TRIPLE FS0; PSID
060540    *"
"060540     INBX21:
060540    *"8C1X2+8C2X2+8C3X2+8C4X2+8C5X2+8C6X2
"060540             X:="ADRX21"=:CINDX
060542            DO WHILE CINDX<<"EAX21"
060546               X:="SG41".BPAGLINK; T:=CORMBANK; *LDDTX DPGPR
060552               A:=0; AD SHZ 12=:DCAD
060555               TAD:=CINDX.FS0; X:=T
060560               A:=;D-D-1 SH 1=:X.MAX
060565               A:=D-"XSPT3"+16BADR=:X.BUFST
060571               CADR=:X.MASTB; CINDX+3=:CINDX
060576            OD
060577    *
"060577             EXIT
060600    RBUS
```

```
060605
060605
060605    *"8D023
"060605
060605    %=========================================================================
060605    %  34.36      E D T R M
060605    %
060605    % MONITOR CALL TO ENABLE/DISABLE PROGRAM TERMINATION HANDLING
060605    %
060605    % MONITOR CALL (206): ISTAT=EDTRM(<ENABLE/DISABLE>,<RT/UB/FE>
060605    %                                 < 1   /   0   >,< 0/ 1/ 2>
060605    %
060605    %
060605
060605    SUBR EDTRM
060605
060605    EDTRM: CALL GET2
060606           IF BACKGROUND><0 THEN
060610                 IF D1=0 GO ILLPA
060612           ELSE
060613                 IF D1>0 GO ILLPA
060616           FI
060616           IF D1=0 THEN           % RT
060620                 IF D0=0 THEN     % DISABLE
060622                    0=:FLRTTERM
060623                 ELSE             % ENABLE
060624                    IF RTTERM=0 GO ILLPA
060626                    1=:FLRTTERM
060630                 FI
060630           ELSE                   % BACKGROUND
060631                 MLEV; *MST PIE  % ENABLE MONITOR LEVEL
060633                 T:=50PSEG; X:="PETECOM"; CALL GET1L; GO ERR
060637                 IF A="TECBUF" GO ILLPA
060642                 IF D1 BIT "0" THEN % USER-BREAK
060645                    IF D0=0 THEN   % DISABLE
060647                       FLBGTERM BZERO "0"=:FLBGTERM
060652                    ELSE           % ENABLE
060653                       FLBGTERM BONE "0"=:FLBGTERM
060656                    FI
060656                 FI
060656                 IF D1 BIT 1 THEN % FATAL-ERROR
060661                    IF D0=0 THEN   % DISABLE
060663                       FLBGTERM BZERO 1=:FLBGTERM
060666                    ELSE           % ENABLE
060667                       FLBGTERM BONE 1=:FLBGTERM
060672                    FI
060672                 FI
060672           FI; A:=0; GO OUT
060674
060674    ERR:   -1; GO OUT              % ERROR WHEN ACCESSING OPCOM SEGMENT
060676    ILLPA: 174                     % ILLEGAL PARAMETER
060677    OUT:   A=:ZAREG; GO RET
060701    RBUS
060712
060712
060712    %=========================================================================
060712    %  34.37      M P A S E T   M P A G E T
060712    %
060712    %  MONITOR CALL TO SET (56) AND GET (57) USER PARAMETERS
060712    %  CALLABLE FROM BACKGROUND ONLY
```

```
060712   %
060712   %  IERR=PASET(APAR),  IERR=PAGET(APAR)
060712   %
060712
060712   SUBR MPASET,MPAGET
060712
060712   DISP 0; REAL FS0=S0; DOUBLE DIS3=S3; PSID
060712
060712   MPASET: K:=1; GO FSGPAR
060714   MPAGET: K:="0"
060715   FSGPAR: T:="P0"; CALL GET0; T=:D0
060720           IF DEMAND><0 THEN MLEV; *MST PIE
060724           FI
060724           IF K THEN                % SET PARAMETERS
060726                X:=D0; CALL ALTON; X.FS0; CALL ALTOF; TAD=:"USPAR".FS0
060734                X:=D0; CALL ALTON; X.DIS3; CALL ALTOF; AD=:"USPAR".DIS3
060742           ELSE                     % GET PARAMETERS
060743                "USPAR".FS0; X:=D0; CALL ALTON; TAD=:X.FS0; CALL ALTOF
060751                "USPAR".DIS3; X:=D0; CALL ALTON; AD=:X.DIS3; CALL ALTOF
060757           FI; A:=0
060760   OUT:    A=:ZAREG; GO RET
060762
060762   RBUS
060770
060770
060770   %=============================================================================
060770   %  34.38      R E R R P
060770   %
060770   %  MONITOR CALL (207) TO READ ERROR PARAMETERS (9ERRP+ABPRO)
060770   %
060770   %  CALL RERRP(IARR)
060770   %
060770   %       IARR(1) = ERNUMB (ASCII)
060770   %       IARR(2) = ERPREG
060770   %       IARR(3) = N1 (A-REG)
060770   %       IARR(4) = N2 (T-REG)
060770   %       IARR(5) = RTPROG
060770   %       IARR(6) = ABPRO (0 IF ABORTED BY SYSTEM, ELSE RTPROG)
060770   %
060770
060770   SUBR RERRP
060770
060770   DISP 0; REAL FS0=S0,FS1=S3; PSID
060770
060770   RERRP: T:="P0"; CALL GET0; T=:D0
060773          IF DEMAND><0 THEN MLEV; *MST PIE
060777          FI; RTREF=:D; D0; T:=6; CALL CHLIM; GO ILLAD
061005          "9ERRP".FS0; X:=D0; CALL ALTON; TAD=:X.FS0; CALL ALTOF
061013          "9ERRP".FS1; X:=D0; CALL ALTON; TAD=:X.FS1; CALL ALTOF
061021          A:=0
061022   OUT:   A=:ZAREG; GO RET
061024   ILLAD: 153; GO OUT             % ILLEGAL ADDRESS REF. IN MONITOR CALL
061026
061026   RBUS
061036
061036
061036
061036   *  8NSDB
061036   %=============================================================================
061036   %  34.39      B R P N T   D E B U G
```

```
061036   %
061036   %  SPECIAL MONITOR CALLS FOR NORD SYMBOLIC DEBUGGER (MON 204 AND 205)
061036   %
061036
061036   *)9RLPL

% @MAC
)9SCLC
061036  MAXUS=0
061036   8DB01;)KILL MAXUS;MAXUS=1
061036   8NSDB
061036  )9RCLC
%
% *)9SLPL
)9SLPL061036   INTEGER NMAXUSERS; * *-1/MAXUS
061037   INTEGER ARRAY BUFFR(7)
061046   INTEGER CUIDX
061047   * *<* MAXUS
061047   *)ZERO
061047   INTEGER ARRAY DBGPROGS(0); * *+MAXUS/
061050
061050   SUBR BRPNT,DEBUG,ESCDEBUG,3BRPNT,3DEBUG
061050
061050   BRPNT: CALL GET0; MLEV; *MST PIE
061053          CALL S3NOALPIT; GO 3BRPNT
061055   DEBUGGER: CALL GET0; MLEV; *MST PIE
061060          CALL S3NOALPIT; GO 3DEBUG
061062
061062   % CALLED FROM MESCAPE
061062   INTEGER POINTER ELREG
061063   ESCDBUG: X;=L=:"ELREG";=B; T:="DBPROG"; CALL XGTDFADDR
061070            FOR X:=0 TO NMAXUSERS-1 DO
061075               IF T;=DBGPROGS(X)=A THEN 0=:DBGPROGS(X); GO ELREG FI
061102            OD; GO ELREG
061105   RBUS
061115
061115   *"
"061115
061115   *"8VDD+8VDP
"061115   %=============================================================================
061115   %
061115   % 34.40 ***************** VIRTUAL DISK DRIVER ROUTINES ************************
061115   *"8VDD+8VDP
"061115   @LIB OLDPIOC
061115   @DEV 1
061115   @DEV (S-S-J)XMSG-SYSTABS
```

```
061115
061115  %=============================================================================
061115  %       X M S G - S Y S T A B S
061115  %=============================================================================
061115
061115
061115  %=============================================================================
061115  % 34.40       X M S G - S Y S T A B S
061115  %
061115  %
061115  % DEFINITIONS FOR TABLES COMMON TO BOTH RESIDENT AND POF CODE
061115  %
061115  %*****************************************************************************
061115
061115  % DEFINE THE INTERRUPT LEVEL THAT XMSG WILL USE
061115  SYMBOL 5XLEV=5                  % INTERRUPT LEVEL FOR RUNNING DRIVER MONITOR
061115  SYMBOL XLEVL=5XLEV,XLEVB=10*5XLEV       % DEDUCTIONS FROM THE ABOVE
061115  *XLEV=1@5XLEV                           % BIT MASK FOR MST(CL) PID(E)
061115
061115  % DESCRIPTION OF AN XT-BLOCK:
061115
061115  DISP 5TLEN=0                    % TASK DESCRIPTOR BLOCK (EXTENSION OF RT-DESCR)
061115         INTEGER XTCHN            % CHAIN FOR FREE/WAITING FOR RESOURCES
061115         INTEGER XTSTA            % STATUS WORD
061115         INTEGER XTRTA            % RT-ADDRESS OR 0 IF DRIVER
061115         INTEGER XTPRT            % PORT CHAIN HEADER (POINTS TO FIRST IN QUEUE)
061115         INTEGER XTMEM            % NUMBER OF BYTES OF MEMORY IN USE
061115         INTEGER XTMMX            % MAX AMOUNT OF MEMORY ALLOWED
061115         INTEGER XTCMS            % TASK CURRENT MESSAGE
061115         INTEGER XTSTS            % BIT 17: PAGING STATUS, 0-4: INTERRUPT LEVEL
061115         INTEGER XTPRG,XTXRG,XTTRG,XTARG,XTDRG,XTLRG,XTSRG,XTBRG % SAVE AREA
061115         DOUBLE XTMRG=XTARG               % AD REGS
061115         TRIPLE XTFRG=XTTRG               % TAD REGS
061115         INTEGER XTAPR            % SAVED ACTPRI WHILE ON LEVEL 1 IN XMSG (YSTL1)
061115         INTEGER XTTAP            % ACTPRI TO USE TO SET UP WINDOWS FOR TRANSFER
061115         INTEGER XTUBF            % USER BUFFER ADDRESS
061115         INTEGER XTSBK, XTSBF     % SYSTEM BUFFER BANK/ADDR. BIT 15=>USER->SYS
061115         DOUBLE XTSAD=XTSBK       % COLLECTIVE NAME
061115         INTEGER XTCNT            % TRANSFER COUNT IN BYTES
061115         INTEGER XTTCN            % DYNAMIC COUNTER USED BY XTRAN
061115         INTEGER POINTER XTHOM % RETURN ADDRESS FROM XTRAN. ALSO USED BY XDHAN!
061115         % SAVE AREA FOR SEGMENT INFORMATION DURING INDIRECT TRANSFER
061115         INTEGER XTASG            % ACTUAL SEGMENTS 1 AND 2
061115         INTEGER XTRSG            % RE-ENTRANT SEGMENT
061115         INTEGER XTBM0,XTBM1      % BIT MAP
061115         INTEGER XTBM2,XTBM3
061115         TRIPLE XTBMA=XTRSG; DOUBLE XTBMB=XTBM2
061115  PSID
061115  % BITS IN XTSTA ARE NOW DEFINED IN XMSG-POFTABS
061115  %*****************************************************************************
061115  %       END OF XMSG-SYSTABS:SYMB
061115  %*****************************************************************************
061115
061115  !DEV 1
061115  %DEV (S-S-J)CDR3
```

```
061115
061115   %=============================================================================
061115   %       C D R 3
061115   %=============================================================================
061115
061115
061115   *'8ACC                                    % CPU-TIME USED FOR USER RT-PROGRAMS
"061115   %=============================================================================
061115   % 34.50       N A C C O U N T
061115
061115   SUBR NACCOUNT
061115   %% RT-ACCOUNTING                                  % USER RT-PROGRAM
061115   NACCOUNT:   IF A:=CURPROG>"RTBES" GO NN2
061121           IF     A="RWRT1" THEN A:="DF1".SSREF;GO NN1;FI
061127           IF     A="RWRT2" THEN A:="DF2".SSREF;FI
061134   NN1:    IF     A<"RTBES" THEN EXIT;FI            % NOT USER RT-PROGRAM
061140   NN2:           A-"RTBES"=:D:=0;T:=5RTSIZE;*RDIV ST % COMPUTE RT-DES INDEX
061145           A=:D SHZ  1 +D+ "ACTAB"=:X               % ACCOUNT ENTRY
061152           IF X.S0 >< 0 THEN
061154             MIN X.S2;GO SUITE;MIN X.S1;0/\0
061160           FI
061160   SUITE:  EXIT
061161   *)FILL
061170   *"-8ACC
"061170   RBUS
061170
061170   INTEGER BPAG1=?                          %% POINTER TO 1. ELEM. IN SEG.TABLE
061170   INTEGER CCSTART=?                        %% RTCOMMON START IN PHYSICAL MEMORY
061170   *"8RSEG+8N500
"061170   %=============================================================================
061170   % 35.0        M R S E G M
061170   %
061170   %       MONITOR CALL TO READ SEGMENT TABLE ENTRY
061170   %
061170   %       ENTRY:  D0 = SEGMENT NUMBER
061170   %               D1 = USER ARRAY ADDRESS WHERE CONTENT SHOULD BE STORED
061170   %
061170   %       EXIT :  OK = SKIPRETURN
061170   %               ERROR:  A=ERRCODE
061170
061170   SUBR MRSEGM
061170
061170   INTEGER ARRAY SCCOM(5)                           % SIM. SEGMENT TABLE ENTRY FOR RT-COMMON
061175   DISP 0; REAL FS0=S0; DOUBLE DIS3=S3; PSID
061175
061175   MRSEGM: T:="P1"; CALL GET1; T=:D1
061200           IF D0<0 OR A>SGMAX GO ILLPA
061205           A*5SEGSIZE+SEGSTART=:D2
061210           RTREF=:D; D1; T:=5; CALL CHLIM; GO ILLAD
061216           IF DEMAND><0 THEN MLEV; *MST PIE
061222           FI
061222           IF D0=0 THEN            % RT-COMMON
061224                 100-CCFPAGE SH 10+CCFPAGE=:SCCOM(2)
061232                 CCSTART=:SCCOM(3)
061235                 161000=:SCCOM(4); "SCCOM"=:D2
061242           FI
061242           D2.FS0; X:=D1; CALL ALTON; TAD=:X.FS0; CALL ALTOF
061250           D2.DIS3; X:=D1; CALL ALTON; AD=:X.DIS3; CALL ALTOF
061256           MIN ZPREG; 0/\0; A:=0
061261   OUT:    A=:ZAREG; GO RET
```

```
061263    ILLPA: 174; GO OUT            % ILLEGAL PARAMETER
061265    ILLAD: 153; GO OUT            % ILLEGAL ADDRESS REF. IN MON CALL
061267    RBUS
061304    *"-8RSEG -8N500
"061304
061304    %=============================================================================
061304    % 35.2        R T D I R
061304    %
061304    %SUBROUTINE TO START AN RT PROGRAM FROM A DIRECT TASK
061304    %A=POINTER TO AN ARRAY OF 5 LOCATIONS, THE FIRST=RT PROGRAM
061304    SUBR RTDIR
061304    DISP 7; INTEGER AREG,LREG; PSID
061304    RTDIR: B:=:A-4; A=:AREG:=L=:LREG
061311           "RTDMON"=:MFUNC; CALL RTACT
061314           LREG=:L; AREG:=:B+4; EXIT
061322    %MONITOR LEVEL:
061322    RTDMON: X.ISTATE; IF =0 THEN CALL 9ERR(#01) FI; CALL XRTCHECK
061327           CALL RTENTRY; GO STUPR
061331    RBUS
061337
061337
061337    %=============================================================================
061337    %
061337    % 35.3        P R T D R
061337    %
061337
061337    SUBR PRTDR
061337    PRTDR: IF X:="RTDTTABLE"=0 GO OUT
061341           DO WHILE X.S1><0
061343                  IF "ERTDTABLE"=X GO OUT
061346                  X+5
061347           OD; T=:X.S0; X=:A;  CALL RTDIR
061353    OUT:   GO RET14
061354    RBUS
061360
061360    *"
"061360
061360
```

```
061360
061360    *'8DIR
"061360
061360    SUBR P3OSTERM
061360    RBUS
061360
061360    %=============================================================================
061360    % 35.9        3 I N S T R   3 O U T S T
061360    %
061360    % STRING MONITOR CALLS
061360    % I=INSTR(LOG.NO,COREADDR,MAXNO,TERMINATOR)
061360    % I=OUTST(LOG.NO,COREADDR,NUMBER)
061360    SUBR 3INSTR,3OUTST,PT3OSTERM,COPTDFIELD
061360
061360    DISP 24
061360            INTEGER CCOUNT,VAL,DYTYFIELD
061360            INTEGER POINTER LREG
061360    PSID
061360
061360    @LIB CXCPU
061360    INTEGER POINTER IPITO:=177000+5BFPAGE+5BFPAGE
061361    DOUBLE POINTER DPITO=IPITO
061361    1SETBFPAGE: A:=TDFPHPAGE=:D:=162000; *POF
061365                AD=:DPITO; *PON
061367                A:=TDFLGADDR/\1777+"5BFPAGE*2000"=:B
061373            EXIT
061374    @ELIB
061374
061374    INTEGER BREG
061375
061375    % MONITOR CALL INSTR:
061375    3INSTR: T:="P1"; CALL GET4; CALL C3OUTX
061400            A=:X; CALL CHERR
061402            OLDPAGE=:D; CALL DALTON; *BSET ZRO
061406            DO
061406                  MIN CCOUNT; GO BYP1; GO OUT1
061411    BYP1:         CALL IOTR; GO WEX1; IF =D3 GO TERM1; A SH 10=:VAL
061420                  MIN CCOUNT; GO BYP2; GO OUT2
061423    BYP2:         CALL IOTR; GO WEX2; IF =D3 GO TERM2
061430                  A+VAL; *BSET ONE
061432                  A=:X.S0; *BSET ZRO
061434                  X+1
061435            OD
061436    %EXITS FROM LOOP:
061436    OUT2:   *BSET ONE
061437            A=:X.S0; *BSET ZRO
061441    OUT1:   D2=:ZAREG; GO ACTRET
061444    % BUFFER EMPTY(RT ONLY):
061444    WEX2:   VAL; *BSET ONE
061446            A=:X.S0; *BSET ZRO
061450            GO WEXX
061451    WEX1:   IF D2+CCOUNT-1=0 GO WTIO
061455    WEXX:   D2+CCOUNT BONE 17=:ZAREG; GO ACTRET
061462    TERM2:  A+VAL; GO TERMX
061464    TERM1:  A SH 10
061465    TERMX:  *BSET ONE;
061466            A=:X.S0; *BSET ZRO
061470            D2+CCOUNT+1 BONE 16=:ZAREG; GO ACTRET
061476    % EXIT WITH ACTIVATION OF DEVICE:
061476    ACTRET: X:=DYTYFIELD:=:B; *IOF
```

```
061501    @LIB CXCPU
061501           IF TYPRING BIT 5TERM THEN CALL 1SETBFPAGE FI
061505           CALL STDEV; X;=:B; *PIOF
061510           0=:IPITO; *PION
061512    @ELIB
061512    @LIB CXCPU-,
061512    RETU:   IF BACKGROUND><0 THEN CALL XBMRET FI; GO RET
061516    WTIO:   *IOF
061517           X:=DYTYFIELD:=:B=:BREG:=RTREF; CALL WDATA
061524    @LIB CXCPU
061524           IF TYPRING BIT 5TERM THEN CALL 1SETBFPAGE FI
061530           CALL STDEV; BREG=:B; *PIOF
061534           0=:IPITO
061535    @ELIB
061535    @LIB CXCPU-,
061535           ZPREG-1=:ZPREG; GO RETSTUPR
061541    @LIB CXCPU-,
061541    *)FILL
061557
061557
061557
061557    % MONITOR CALL OUTST;
061557    3OUTST: T:="P1"; CALL GET3; CALL C3OUTX
061562           IF D=0 GO ERR2
061564    @LIB CXCPU
061564           IF D.TYPRING BIT 5TERM GO FAR OSTERM
061570           IF A BIT 5COM GO FAR SIMMAG
061572           IF A BIT 5BAD GO FAR OSTAD
061574    @ELIB
061574    @LIB CXCPU-,
061574           IF BACKGROUND=0 THEN
061576              IF D2>>X.MAX GO OUTFULL
061602              IF A>>X.CFREE GO WTIO
061605           FI; CALL CHERR; OLDPAGE=:D; CALL DALTON; *BSET ZRO
061612           DO
061612                 MIN CCOUNT; GO OBYP1; GO OOUT
061615    OBYP1:       *BSET ONE
061616                 X.S0; *BSET ZRO
061620                 X+1; A=:VAL SHZ -10
061623                 CALL IOTR; GO OUTFULL; MIN CCOUNT; GO OBYP2; GO OOUT
061630    OBYP2:       VAL/\377; CALL IOTR; GO OUTFULL
061634           OD
061635    % EXITS FROM LOOP;
061635    OOUT:  0=:ZAREG; GO ACTRET
061637    OUTFULL: 100000=:ZAREG; GO RET                    % RT ONLY
061642
061642    % LOCAL SUBROUTINE TO CHECK FOR ERRORS; X=DATAFIELD
061642    %RETURN: X=CORE ADDR
061642    CHERR: A:=L=:"LREG"; X=:DYTYFIELD
061645           IF X=0 OR X.TYPRING NBIT 5IOBT GO ERR2
061651           IF X.RTRES><RTREF THEN 5; GO ERRF FI; T=:D
061660           IF D2<0 GO ERR; T:=A+1 SHZ -1; A-,=:CCOUNT
061666           D1; CALL CHLIM; GO ERR; *IOF
061672           IF X.TYPRING BIT 5TERM THEN
061675              T:="FLAGB"; CALL XGTDFADDR
061677              IF A BIT 5LSTA THEN TER02; GO ERRF FI  % TERMINAL NOT CONNECTED
061703           FI
061703    @LIB CXCPU
061703           T:="DERROR"; CALL XGTDFADDR
061705           IF A><0 THEN
```

```
061706              A\/140000=:ZAREG
061710              A:=0; T:="DERROR"; CALL XSTDFADDR
061713              GO FAR RETU
061714    @ELIB
061714    @LIB CXCPU-,
061714           FI; X:=D1; MLEV; *MST PIE; ION
061720           GO LREG
061721    @LIB CXCPU
061721    ERR2:   A:=2
061722    ERRF:   A\/140000; GO FERR
061724    ERR:    A:=-1
061725    FERR:   A=:ZAREG; GO FAR RETU
061727    @ELIB
061727    @LIB CXCPU-,
061727    *)FILL
061754
061754    CBOUTX: T=:D1:=L=:"LREG"
061757            CALL 3OUTX; A=:D; IF A/\177700=100 GO FAR ERR2; A:=D
061766            CALL LOGPH; GO LREG
061770
061770    % SUBROUTINE TO EXECUTE "IOTRANS"
061770    % RETURN: FULL/EMPTY (RT ONLY)
061770    % SKIPRETURN: OK
061770    IOTR:   A=:D; DYTYFIELD; *IOF; IRW BLEVB DB
061774           "BIOTR"; *IRW BLEVB DP
061776           BLEV; *MST PID; COPY SD DA
062001           IF T:=BACKGROUND><0 THEN L-1 FI
062005           *ION; EXIT
062007    % INBT-OUTBT LEVEL:
062007    BIOTR: *PIOF
062010    @LIB CXCPU
062010           IF TYPRING BIT 5TERM THEN
062013              CALL SETBFPAGE; *PON
062015           FI
062015    @ELIB
062015           *IRR ALEVB DA
062016           CALL IOTRANS; GO WWT; *IRW ALEVB DA
062021           *IRR ALEVB DL; AAA 1
062023           IF T:=BACKGROUND><0 THEN A+1 FI
062027           *IRW ALEVB DL
062030    WBIO:   CALL STDEV
062031    @LIB CXCPU
062031           A:=42; *IRR PCR
062033    @ELIB
062033           *PION; WAIT
062035           CALL ERRFATAL
062036    WWT:    IF BACKGROUND><0 THEN
062040    @LIB CXCPU
062040              X:=RTREF
062041              IF TYPRING BIT 5TERM THEN
062044                 TDRADDR; A=:D; D:=:B; CALL WDATA; B:=D
062051              ELSE
062052                 CALL WDATA
062053              FI; "STUPR"; *IRW MLEVB DP
062055    @ELIB
062055    @LIB CXCPU-,
062055              MLEV; *MST PID
062057           FI
062057           GO WBIO
062060    *)FILL
```

```
062073
062073    %       O S T E R M
062073
062073    DISP 0; DOUBLE CDBUADR=CBUADR, DD1=D1; PSID
062073    DISP 5REG; REAL ZF1RG,ZF4RG,ZF7RG; PSID
062073    INTEGER XREG,SAVB
062075
062075    OSTERM: IF D.RTRES><RTREF THEN 5; GO FAR ERRF FI          % DEVICE NOT RESERVED
062104    @LIB CXCPU
062104            *PIOF
062105            GO P3OSTERM                                     % CONTINUE ON PIT 3
062106
062106    COPTDFIELDS: *PION
062107            AD:=DD1=:X.CDBUADR
062111            TAD:=ZF1RG=:X.ZOPRG:=ZF4RG=:X.ZOARG:=ZF7RG=:X.ZOSRG
062117            A:=OLDPAGE
062120            *PIOF
062121            EXIT
062122
062122    @ELIB
062122
062122    @LIB CXCPU-,
062122
062122    TGORET: CALL ALTOFF; *IOF
062124            CALL STDEV
062125            A:=B+"ZOPRG"-5REG=:B; 0=:ZAREG
062132            *PION
062133            GO FAR RETU
062134
062134    @LIB CXCPU-,
062134
062134    INTEGER POINTER PTTNO:=TTNO,PBCHFLAG:=BCHFLAG, PTTIFIELD:=TTIFIELD
062137    30UTX: IF BACKGROUND><0 THEN
062141                X:=B+7; *LRB BLEVB
062144                X:="ESCBLOCK"; *SRB BLEVB
062146                A:=B=:CMDFFIELD
062150                IF D0=1 THEN
062154                    IF PBCHFLAG=0 THEN
062156                        A:=PTTNO
062157                    ELSE
062160                        A:=PTTIFIELD.DFOPP.ROFIL
062163                FI;FI
062163            ELSE
062164                D0
062165            FI; EXIT
062166
062166
062166    % SIMULATE MAGTP CALL FOR COMM. CHANNEL
062166    SIMMAG: D2 SHZ -1=:IMAXW; D1=:ICORAD; IF M THEN MIN IMAXW; 3 ELSE 1 FI A=:IFUNC
062202            1=:WFLAG; 0=:MTFLG; -1=:IBLOADR; GO FAR XMRW              % CONT. IN MAGTP ROUTINE
062210    *)FILL
062222
062222    %       O S T A D
062222
062222    INTEGER TADDF
062223    OSTAD:
062223
062223    *'8STRN+8N500 -BADAD
062223
062223            IF D.RTRES><RTREF THEN 5; GO FAR ERRF FI          % DEVICE NOT RESERVED
```

```
062232          X=:TADDF; D1=:X.USDADR; D2-=:X.OBCOU
062240          TAD:=ZF1RG=:X.ZOPRG:=ZF4RG=:X.ZOARG:=ZF7RG=:X.ZOSRG
062246          IF B="DEMFIELD" THEN X:=RTRES; CALL BRELEASE FI
062253          OLDPAGE=:D; TADDF=:B; MLEV; *MST PIE
062261          0=:XRSA; CALL DALTON
062263  TLOOP:  *PIOF
062264          X:=XRSA; IF OBCOU=0 GO FAR TGORET
062270          FOR OBCOU DO
062270                T:=USDADR
062271                *PION; BSET ONE; LBYT; BSET ZRO; PIOF
062276                CALL IOTRANS; GO TBUFWT; X+1
062301          OD; GO FAR TGORET; *)FILL
062312
062312  TBUFWT: IF DERROR=0 THEN                          % WAIT FOR BUFFER
062314             X:=RTRES; CALL WDATA
062316             "STUPR"; *IRW MLEVB DP
062320             MLEV; *MST PID; PION
062323             GO TLOOP
062324          FI; *PION
062325          0=:DERROR; GO FAR ERRF                    % ERROR FROM IOTRANS
062327  * 8STRN+8N500
"062327   RBUS
062332
062332  @LIB CXCPU
062332  INTEGER ARRAY 30WFIELD(11)              % MONITOR CALL WORKING FIELD FOR OUTSTRING
062343  @ELIB
062343  *"-8STRN -8N500
"062343   *CBLOC=TEXIT; CB2LO=TEXIT; CBUNL=TEXIT
062343
062343  * 8DMAS
"062343   *"8MEAS
"062343   %=============================================================================
062343  % 35.12       X P P R U T
062343  %
062343  SUBR XPPRUT
062343  DISP 7; INTEGER P7; PSID
062343  XPPRUT: *IOF
062344          X:=X.P7=:B; CALL STDEV; *ION
062350          GO STUPR
062351  RBUS
062352
062352
062352  * 7NDDI
```

```
062352    *
062352    * 8QBPO
062352
062352
062352    %=======================================================================
062352    %=======================================================================
062352    %                                                                      %
062352    %                                                                      %
062352    %                                                                      %
062352    %            B U F F E R   P O O L    M O N I T O R   C A L L S     %
062352    %                                                                      %
062352    %                                                                      %
062352    %=======================================================================
062352    %=======================================================================
062352
062352
062352
062352
062352
062352
062352    SYMBOL SBMAX=100              % MAX NUMBER OF BUFFERS IN PACKET
062352    %
062352    %=======================================================================
062352    %          DATA FIELD DESCRIPTORS
062352    %
062352    %=======================================================================
062352    % 35.21      P O O L   H E A D
062352
062352    DISP -1
062352    INTEGER WPVER                 % VERIFICATION VALUE OF POOL HEAD
062352    SYMBOL SPCOR=125252           % CORRECT VALUE OF WPVER
062352    INTEGER WFREE                 % ADDRESS OF FIRST FREE BUFFER, =0 IF EMPTY
062352    INTEGER WNFRE                 % NUMBER OF FREE BUFFERS IN POOL
062352    INTEGER WLLIM                 % LOWER ADDRESS OF BUFFER POOL
062352    INTEGER WHLIM                 % HIGHER ADDRESS OF BUFFER POOL
062352    INTEGER WBFSZ                 % BUFFER SIZE , NO. OF WORDS
062352    PSID
062352
062352
062352    %-----------------------------------------------------------------------
062352    % 35.22      Q U E U E   H E A D
062352
062352    DISP 0
062352    INTEGER WLPAK                 % ADDRESS OF LAST PACKET IN QUEUE
062352                                  % EQUAL TO QUEUE HEAD IF QUEUE EMPTY
062352    INTEGER WFPAK                 % ADDRESS TO FIRST BUFFER IN QUEUE
062352                                  % EQUAL TO QUEUE HEAD IF EMPTY
062352    INTEGER WPNUM                 % NUMBER OF PACKETS IN QUEUE
062352    INTEGER WPROC                 % ADDRESS OF OWNER OF QUEUE
062352    PSID
062352
062352
062352    %-----------------------------------------------------------------------
062352    % 35.23      P A C K E T   H E A D
062352
062352    %       2.3.1 GENERAL OFFSETS, PACKET HEAD AND WXX
062352    %             TO BE USED WITHIN GLOB-DEF, ONLY !!!
062352
062352    SYMBOL SLHW=400               % MULTIPLIER FOR LEFT HALFWORD
062352    SYMBOL S1QW=10000             % MULTIPLIER FOR 1 QUARTERWORD
062352    SYMBOL S3QW=20                % MULTIPLIER FOR 3 QUARTERWORD
```

```
062352    SYMBOL S1OCT=40000              % MULTIPLIER FOR 1 OCTET
062352    SYMBOL S3OCT=2000               % MULTIPLIER FOR 3 OCTET
062352    SYMBOL S5OCT=100                % MULTIPLIER FOR 5 OCTET
062352    SYMBOL S7OCT=4                  % MULTIPLIER FOR 7 OCTET
062352    SYMBOL SRHW=1                   % MULTIPLIER FOR RIGHT HALFWORD
062352
062352    %-----------------------------------------------------------------------
062352    % 35.24        GENERAL FIELDS, PACKET HEAD AND WDW'S
062352    %               TO BE USED WITHIN GLOB-DEF ONLY !!!
062352    %
062352    SYMBOL SRHWMASK=377             % RIGHT HALFWORD MASK
062352    SYMBOL SLHWMASK=-SRHWMASK-1     % LEFT HALFWORD MASK
062352    SYMBOL S1QWMASK=17*S1QW         % 1. QUARTER WORD MASK
062352    SYMBOL S2QWMASK=17*SLHW         % 2. QUARTER WORD MASK
062352    SYMBOL S3QWMASK=17*S3QW         % 3. QUARTER WORD MASK
062352    SYMBOL S4QWMASK=17*SRHW         % 4. QUARTER WORD MASK
062352    SYMBOL C1QWMASK=-S1QWMASK-1
062352    SYMBOL C2QWMASK=-S2QWMASK-1
062352    SYMBOL C3QWMASK=-S3QWMASK-1
062352    SYMBOL C4QWMASK=-S4QWMASK-1
```

```
062352    %
062352    %=============================================================================
062352    % 35.25       ACTUAL PACKET HEAD LAY-OUT
062352    %
062352    DISP SPDISP=0
062352    INTEGER WNBUF                    % ADDRESS OF NEXT BUFFER IN PACKET
062352    INTEGER WNPAK                    % ADDRESS OF NEXT PACKET IN QUEUE
062352    INTEGER WPOSS                    % ADDRESS OF OWNER OF PACKET
062352    INTEGER WDUM                     % DUMMY
062352    INTEGER WTYP,WSCOM=WTYP                                % WORD FOR MESSAGE TYPE &  FORMAT
062352    INTEGER WBC,WSEQ=WBC                                   % WORD FOR BYTE COUNT & SEQUENCE CONTROL
062352    INTEGER WDEA,WPRI=WDEA                                 % DESTINATION ADDRESS AND PRIORITY
062352            SYMBOL SPRIMASK=3*S1OCT                        % MSG PRIORITY
062352            SYMBOL CPRIMASK=-SPRIMASK-1
062352            SYMBOL SPRISHIFT=-16
062352            SYMBOL CPRISHIFT=-SPRISHIFT
062352    DOUBLE  WDADR=WDEA                                     % BOTH ADDRESSES
062352    INTEGER WSCA                                           % SOURCE ADDRESS
062352    PSID
062352    *)KILL SLHW S1QW S3QW S1OCT S3OCT S5OCT S7OCT SRHW
062352    *)KILL SRHWM SLHWM S1QWM S2QWM S3QWM S4QWM
062352    *)KILL C1QWM C2QWM C3QWM C4QWM
```

```
062352
062352
062352
062352    % TIME USED BY SINTRAN ROUTINES:
062352    %       GETO - 52MICS (10-54)    25MICS (10-S-02)
062352    %       ALTON- 68MICS (10-54)    33MICS (10-S-02)
062352    %       ALTOF- 56MICS (10-54)    26MICS (10-S-02)
062352
062352
062352
062352
062352    %=======      G E T F          170
062352    %
062352    % GET A NUMBER OF BUFFERS FROM FREE POOL
062352    % SET RT-PROG ADDRESS IN WPOSS-WORD OF EACH BUFFER
062352    %
062352    % GETF:       GET PARAMETERS(POOLHEAD,NO-OF-BUFFERS-REQUIRED)
062352    %             IF FALSE POOLHEAD OR ILLEGAL NO-OF-BUFFERS-REQUIRED THEN
062352    %                          NON-SKIP-RETURN(-1)
062352    %             ENDIF
062352    %             IF POOLHEAD.WNFRE<NO-OF-BUFFERS-REQUIRED THEN
062352    %                 NON-SKIP-RETURN(0)
062352    %             ENDIF
062352    %             POOLHEAD WFREE=:FIRST-BUFFER=:BUFFER
062352    %             FOR COUNT:=2 TO NO-OF-BUFFERS-REQUIRED DO
062352    %                 RTREF=:BUFFER.WPOSS; 0=:BUFFER.WNPAK
062352    %                 BUFFER.WNBUF=:BUFFER
062352    %             ENDDO
062352    %             BUFFER.WNBUF=:POOLHEAD.WFREE
062352    %             0=:BUFFER.WNBUF
062352    %             POOLHEAD.WNFRE - NO-OF-BUFFERS-REQUIRED=:POOLHEAD.WNFRE
062352    %             SKIP-RETURN(FIRST-BUFFER)
062352    % ENTRY:      X - ADDRESS OF POOL HEAD
062352    %             A - NUMBER OF BUFFERS WANTED
062352    % EXIT:
062352    %       NON-SKIP                A=-4                POOL DESTROYED
062352    %                               A=-1                ILLEGAL PARAMETER VALUE
062352    %                               A=0                 NOT ENOUGH FREE BUFFERS
062352    %       SKIP                    A = ADDRESS OF FIRST BUFFER IN CHAIN
062352    % TIME USED: 326 MICS + 41.5 MICS*N (10-54)       N=NO. OF BUFFERS
062352    %             159 MICS + 20 MICS * N (10-S-02)
062352    %
062352
062352    %=============================================================================
062352    % 35.30       U S O X
062352    %
062352    SUBR USOX
062352    INTEGER NUM                     % NUMBER OF BUFFERS REQUIRED
062353    INTEGER LVAR                    % LOCAL VARIABLE
062354    INTEGER ICURT                   % ADDRESS OF CURRENT RT-PROG DESCRIPTION
062355    INTEGER NPOLH                   % ADDRESS OF POOL HEAD
062356    INTEGER LLIM                    % LOWER LIMIT OF BUFFER POOL
062357    INTEGER HLIM                    % HIGHER LIMIT OF BUFFER POOL
062360    USOX:   CALL GETO; ZAREG=:NUM; ZXREG=:NPOLH
062365            RTREF=:ICURT
062367            CALL ALTON              % ACCESS ALTERNATIVE PAGE TABLE
062370            NPOLH.WHLIM=:HLIM; X.WLLIM=:LLIM          % SAVE BUFFER POOL LIMITS
062375            IF NPOLH.WPVER><SPCOR OR NUM<=0 OR A>SBMAX THEN A:=-1; GO NSKIP FI
062412            IF X.WNFRE-NUM<0 THEN A:=0; GO NONSK FI   % TOO FEW BUFFERS LEFT
062417            A=:X.WNFRE                                % UPDATE NUMBER OF FREE BUFFERS LEFT
```

```
062420          1=:D
062422          A:=X.WFREE=:LVAR                                % FIRST FREE BUFFER
062424          FOR D TO NUM DO
062427            IF A=:X-LLIM<<0 OR HLIM-,+X>>=0 THEN A:=-4; GO NONSK FI
062442            IF X.WNPAK><X THEN A:=-4; GO NONSK ELSE 0=:X.WNPAK FI
062450            ICURT=:X.WPOSS; A:=X.WNBUF
062453          OD
062455          0=:X.WNBUF
062456          A=:NPOLH.WFREE
062460          CALL ALTOFF; LVAR=:ZAREG; MIN ZPREG; 0/\0; GO RET
062466  NONSK:
062466  NOSKIP: A=:LVAR
062467          CALL ALTOFF; LVAR=:ZAREG; GO RET; *USO=USOX
062473  RBUS
062501
062501
062501
062501
062501
062501
062501
062501  %========     P U T F          171
062501  %
062501  % RETURN A CHAIN OF BUFFERS TO FREE POOL. ALL BUFFERS MUST CONTAIN
062501  % ADDRESS OF CURRENT RT-PROGRAM IN WPOSS-WORD, AND ZERO IN
062501  % WNPAK-WORD. LAST BUFFER MUST HAVE ZERO IN WNBUF-WORD.
062501  % IT IS CHECKED THAT NUMBER OF BUFFERS DOES NOT EXCEED MAX ALLOWED
062501  % NUMBER. THIS PREVENTS INFINITE LOOP.
062501  %
062501  % PUTF:       GET PARAMETERS(POOLHEAD,PACKET)
062501  %             IF FALSE POOLHEAD OR ILLEGAL PACKET THEN NON-SKIP-RETURN(-1) ENDIF
062501  %             IF ANY-BUF-OF-PACKET.WPOSS><CURR-RT-PROG OR PACKET.WNPAK><0 THEN
062501  %                 NON-SKIP-RETURN(0)
062501  %             ENDIF
062501  %             POOLHEAD.WFREE=:LAST-BUFFER-OF-PACKET.WNBUF
062501  %             PACKET=:POLLHEAD.WFREE
062501  %             0=:ALL-PACKET-BUFFERS.WPOSS
062501  %             POOLHEAD.WNFRE+NO-OF-NUFFERS-IN-PACKET=:POOLHEAD.WNFRE
062501  %             SKIP-RETURN
062501  %
062501  %
062501  % ENTRY;      X - ADDRESS OF POOL HEAD
062501  %             A - ADDRESS OF FIRST BUFFER IN CHAIN
062501  % EXIT:
062501  %                              A=-3                TOO LONG CHAIN, DISORDER ?
062501  %       NON-SKIP               A=-2                PACKET NOT WPOSSESSED BY PROGRAM
062501  %                                                  OR PACKET BELONGS TO QUEUE
062501  %                              A=-1                ILLEGAL PARAMETER VALUE
062501  %       SKIP                   BUFFER CHAIN PUT IN POOL OK
062501  % TIME USED: 320 MICS + 52 MICS * N (10-54)        N=NO OF BUFFERS IN PACKET
062501  %           154 MICS + 24 MICS * N (10-S-02)
062501
062501  %=====================================================================
062501  % 35.31       U S 1 X
062501  %
062501  SUBR US1X
062501  INTEGER IPACK                % ADDRESS OF PACKET TO RELEASE
062502  INTEGER LVAR                 % LOCAL VARIABLE
062503  INTEGER ICURT                % ADDRESS OF CURRENT RT-PROG DESCRIPTION
062504  INTEGER NPOLH                % ADDRESS OF POOL HEAD
```

```
062505    INTEGER LLIM                    % LOWER LIMIT OF BUUFER POOL
062506    INTEGER HLIM                    % HIGHER LIMIT OF BUFFER POOL
062507    US1X:   CALL GETQ; ZAREG=:IPACK; ZXREG=:NPOLH
062514            RTREF=:ICURT
062516            CALL ALTON              % ACCESS ALTERNATIVE PAGE TABLE
062517            NPOLH.WLLIM=:LLIM; X.WHLIM=:HLIM
062524            IF NPOLH.WPVER><SPCOR THEN A:=-1; GO NSKIP FI
062533            IF IPACK.WNPAK><0 THEN A:=-2; GO NSKIP FI
062540            1=:D; IPACK=:LVAR       % CHECK BUFFER CHAIN
062544            FOR D TO SBMAX DO
                    LVAR=:LVAR.WNPAK
062552              IF A=:T-LLIM<<0 OR HLIM-.+T>>=0 THEN A:=-1; GO NSKIP FI
062565              IF LVAR.WPOSS-ICURT><0 THEN A:=-2; GO NSKIP
062573              ELSE 0=:X.WPOSS; FI
062575              IF LVAR.WNBUF=:LVAR=0 GO PEND
062601            OD
062603            A:=-3; GO NSKIP
062605    PEND:   X=:T
062606            NPOLH.WFREE=:T.WNBUF
062612            IPACK=:NPOLH.WFREE
062615            NPOLH.WNFRE+D=:X.WNFRE
062621            CALL ALTOFF; MIN ZPREG; 0/\0; GO RET
062625    NSKIP:  A=:LVAR
062626            CALL ALTOFF; LVAR=:ZAREG; GO RET; *US1=US1X
062632    RBUS
062640
062640
062640
062640
062640
062640    %========   G E T Q          172
062640    %
062640    % GET A PACKET FROM START OF QUEUE
062640    % SET RT-PROG ADDRESS IN WPOSS-WORD OF EACH BUFFER
062640    %
062640    % GETQ:        GET PARAMETER(QUEUEHEAD)
062640    %              IF ILLEGAL QUEUE HEAD THEN NON-SKIP-RETURN(-1) ENDIF
062640    %              IF QUEUE EMPTY THEN NON-SKIP-RETURN(0) ENDIF
062640    %              QUEUEHEAD.WFPAK=:PACKET
062640    %              PACKET.WNPAK=:QUEUEHEAD.WFPAK
062640    %              QUEUEHEAD.WPNUM - 1
062640    %              0=:PACKET.WNPAK
062640    %              QUEUEHEAD.WPROC=:ALL-BUFFERS-OF-PACKET.WPOSS
062640    %              SKIP-RETURN(PACKET)
062640    %
062640    % ENTRY:       X - ADDRESS OF QUEUE HEAD
062640    % EXIT:
062640    %       NON-SKIP                A=-1                    ILLEGAL PARAMETER VALUE
062640    %                               A=0                     QUEUE EMPTY
062640    %       SKIP                    A = ADDRESS OF PACKET
062640    % TIME USED: 281 MICS (10-54)                           134 MICS (10-S-02)
062640
062640    %=============================================================================
062640    % 35.32       U S X 2
062640    %
062640    SUBR US2X
062640    INTEGER IPACK                   % ADDRESS OF FETCHED PACKET
062641    INTEGER ICURT                   % ADDRESS OF CURRENT RT-PROG DESCRIPTION
062642    INTEGER IQH                     % ADDRESS OF QUEUE HEAD
062643    US2X:   CALL GETQ; ZXREG=:IQH; RTREF=:ICURT
```

```
062650          CALL ALTON              % ACCESS ALTERNATIVE PAGE TABLE
062651  GOCOM:  IF IQH.WPROC><ICURT THEN A:=-1; GO NSKIP FI
062660          IF X.WPNUM=0 THEN GO NSKIP FI
062663          A:=X.WFPAK=:IPACK
062665          IF A.WNPAK=IQH THEN IQH=:IQH.WLPAK FI
062675          A=:IQH.WFPAK; X.WPNUM-1=:X.WPNUM
062702          A:=IQH.WPROC; X:=IPACK
062705          DO WHILE X><0
062706            A=:X.WPOSS; 0=:X.WNPAK; X:=X.WNBUF
062711          OD
062712          CALL ALTOFF; IPACK=:ZAREG; MIN ZPREG; 0/\0; GO RET
062720  NSKIP:  A=:IPACK
062721          CALL ALTOFF; IPACK=:ZAREG; GO RET; *US2=US2X
062725  RBUS
062732
062732
062732
062732
062732
062732
062732
062732  %--------     P U T Q          173
062732  %
062732  % PUT A PACKET AT END OF QUEUE. WPOSS-WORD MUST CONTAIN ADDRESS
062732  % OF CALLING RT-PROGRAM. WNPAK-WORD MUST BE ZERO.
062732  %
062732  % PUTQ:       GET PARAMETER(QUEUEHEAD,PACKET)
062732  %             IF PACKET.WPOSS><CURR-RT-PROG THEN NON-SKIP-RETURN(-2) ENDIF
062732  %             IF PACKET.WNPAK><0 THEN NON-SKIP-RETURN(0) ENDIF
062732  %             QUEUEHEAD.WLPAK=:PACKET.WNPAK
062732  %             PACKET=:QUEUEHEAD.WLPAK
062732  %             DO WHILE PACKET><0
062732  %                 0=:PACKET.WPOSS; PACKET.WNPAK=:PACKET
062732  %             ENDDO
062732  %             QUEUEHEAD.WPNUM + 1
062732  %             SKIP-RETURN
062732  %
062732  % ENTRY:      X - ADDRESS OF QUEUE HEAD
062732  %             A - ADDRESS OF PACKET
062732  % EXIT:
062732  %       NON-SKIP              A=-2                  PACKET NOT WPOSSESSED BY PROGRAM OR
062732  %                                                   PACKET BELONGS TO QUEUE
062732  %       SKIP                  PACKET PUT IN QUEUE
062732  % TIME USED:
062732  %       PUTQ: 320 MICS (NORD-10)                    165 MICS (10-S)
062732
062732  %=============================================================================
062732  % 35.33       U S X 3
062732  %
062732  SUBR US3X
062732  INTEGER IPACK               % ADDRESS OF PACKET TO PUT IN QUEUE
062733  INTEGER ICURT               % ADDRESS OF CURRENT RT-PROG DESCRIPTION
062734  INTEGER IQH                 % ADDRESS OF QUEUE HEAD
062735  US3X:   CALL GETO; ZAREG=:IPACK; ZXREG=:IQH
062742          RTREF=:ICURT
062744          CALL ALTON              % ACCESS ALTERNATIVE PAGE TABLE
062745          IF IPACK.WPOSS><ICURT OR X.WNPAK><0  THEN A:=-2; GO NSKIP FI
062756          T:=IQH.WLPAK; IPACK=:X.WLPAK; MIN X.WPNUM; A=:T.WNPAK
062765          IQH=:IPACK.WNPAK
```

```
062770          X:=IPACK
062771          DO WHILE X><0
062772            0=:X.WPOSS; X:=X.WNBUF
062774          OD
062775          CALL ALTOFF; MIN ZPREG; 0/\0; GO RET
063001  NSKIP:  A=:IPACK
063002          CALL ALTOFF; IPACK=:ZAREG; GO RET; *US3=US3X
063006  RBUS
063013
063013
063013
063013
063013
063013
063013  %========     P R I Q          174
063013  %
063013  % PUT A PACKET IN QUEUE ACCORDING TO PRIORITY, I.E. PACKET IS INSERTED
063013  % AFTER LAST PACKET IN QUEUE WITH EQUAL OR HIGHER PRIORITY.
063013  % PRIORITY VALUE IS FOUND IN PACKET HEAD, WORD AND BITS DEFINED AT SYSTEM
063013  % GENERATION.
063013  % WPOSS-WORD MUST CONTAIN ADDRESS OF CALLING RT-PROGRAM. WNPAK-WORD
063013  % MUST BE ZERO.
063013  %
063013  % PRIQ:       GET PARAMETER(QUEUEHEAD,PACKET)
063013  %             IF ANY-BUF-OF-PACKET.WPOSS><CURR-RT-PROG OR PACKET.WNPAK><0 THEN
063013  %                 NON-SKIP-RETURN(-2)
063013  %             ENDIF
063013  %             IF PACKET.PRIOR=0 THEN
063013  %                 QUEUEHEAD.WLPAK=:LASTPACKET; QUEUEHEAD=:QPACK
063013  %                 GO INSERT
063013  %             ENDIF
063013  %             QUEUEHEAD.WFPAK=:QPACK
063013  %             DO WHILE QPACK><QUEUEHEAD AND QPACK.PRIOR>=PACKET.PRIOR
063013  %                 QPACK=:LASTPACKET; QPACK.WNPAK=:QPACK
063013  %             ENDDO
063013  % INSERT:     QPACK=:PACKET.WNPAK; PACKET=:LASTPACKET.WNPAK
063013  %             QUEUEHEAD.WPNUM + 1
063013  %             DO WHILE PACKET><0
063013  %                 0=:PACKET.WPOSS; PACKET.WNPAK=:PACKET
063013  %             ENDDO
063013  %             SKIP-RETURN
063013  %
063013  % ENTRY:      X - ADDRESS OF QUEUE HEAD
063013  %             A - ADDRESS OF PACKET
063013  % EXIT:
063013  %       NON-SKIP              A=-2                PACKET NOT WPOSSESSED BY PROGRAM OR
063013  %                                                 PACKET BELONGS TO QUEUE
063013  %       SKIP                  PACKET INSERTED IN QUEUE
063013  % TIME USED:
063013  %       PRIQ: PRIOR 0         330 MICS (NORD-10)  170 MICS (10-S)
063013  %                   ><0       320+33*N (NORD-10)  165+15*N (10-S)
063013  %                   ><0       130+33*N (NORD-10)   65+15*N (10-S)
063013  %                             N=NO OF PACKETS AHEAD IN QUEUE+1
063013
063013  %=============================================================================
063013  % 35.34       U S X 4
063013  %
063013  SUBR US4X
063013  INTEGER IPACK                 % ADDRESS OF PACKET TO SEND
063014  INTEGER ICURT                 % ADDRESS OF CURRENT RT-PROG DESCRIPTION
```

```
063015    INTEGER IQH                    % ADDRESS OF QUEUE HEAD
063016    INTEGER IPRI                   % PRIORITY OF PACKET
063017    US4X:   CALL GETO; ZAREG=:IPACK; ZXREG=:IQH
063024            RTREF=:ICURT
063026            CALL ALTON             % ACCESS ALTERNATIVE PAGE TABLE
063027            IF IPACK.WPOSS><ICURT OR X.WNPAK><0 THEN A:=-2; GO NSKIP FI
063040            X.WPRI/\SPRIMASK=:IPRI
063043            X:=IQH=:D
063045            IF IPRI=0 THEN X.WLPAK=:D; GO DIPUT FI
063052            DO WHILE X.WNPAK><IQH AND A.WPRI/\SPRIMASK-IPRI>>=0
063064              X:=D.WNPAK=:D
063067            OD
063070    DIPUT:  T:=D.WNPAK; IPACK=:X.WNPAK; T=:A.WNPAK
063076            IF A:=T=IQH THEN IPACK=:IQH.WLPAK FI
063105            MIN IQH.WPNUM
063107            X:=IPACK
063110            DO WHILE X><0
063111              0=:X.WPOSS; X:=X.WNBUF
063113            OD
063114            CALL ALTOFF; MIN ZPREG; 0/\0; GO RET
063120    NSKIP:  A=:IPACK
063121            CALL ALTOFF; IPACK=:ZAREG; GO RET; *US4=US4X
063125    RBUS
063133
```

```
063133    *
063133
063133
063133
063133
063133
063133
063133
063133
063133
063133
063133
063133    *"8F16+8DM01+8DM02+8DM03+8DM04+8DM05+8DM06+8DM07+8DM08+8DM09+8DM10
"063133
063133
063133    %=============================================================================
063133    % 35.36       B I O A C C O U N T
063133    %
063133    %% BLOCK I/O ACCOUNTING ROUTINE
063133    %% ROUTINE CALLED FROM PAGE-TRANSFER ROUTINE IN FILE-SYSTEM-RESIDENT PART
063133    INTEGER RWRT1=?,RWRT2=?
063133    INTEGER DF1=?,DF2=?
063133    SUBR BIOACCOUNT
063133
063133    BIOACCOUNT;
063133    *"-BIOAC
"063133             IF BACKGROUND >< 0 AND ACCFLAG ><0 THEN    % BACKGROUND-ACCOUNTING ON
063137                  A:=CURPROG; GO SUITE
063141               FI
063141               IF RTACCFLAG = 0 THEN EXIT; FI       % RT-ACCOUNTING OFF
063144               IF CURPROG = "RWRT1" THEN A:="DF1".SSREF; GO SUITE; FI
063153               IF         = "RWRT2" THEN A:="DF2".SSREF; GO SUITE; FI
063161               EXIT
063162    SUITE:     A-RTSTART=:D;=0;T;=5RTSIZE;*RDIV ST
063167               A SHZ 1 + "IOACTAB"=:X
063172               *PIOF
063173               MIN X.S1;GO RETU; MIN X.S0; 0/\0
063177    RETU:      *PION
063200               EXIT
063201    *"
"063201     RBUS
063213
063213
063213    *"8XMSG
```

```
063213
063213  %*************************************************************************
063213  %
063213  %       X M S G  -  R E S I D E N T    P A R T
063213  %
063213  %       MARTIN STANDLEY. 12/12/79. INSTALLED: 16/9/80 IN SIII E!
063213  %                        18/04-83 for S III I (XMSG/PIT3)
063213  %
063213  %       THIS PART CONTAINS ALL THEN ROUTINES THAT MUST LIE IN THE
063213  %       0-30K PART IF SINTRAN (I.E. BOTH IN POF AND ON PIT0.)
063213  %
063213  %       LINKAGE BETWEEN THE RESIDENT AND KERNEL(PIT3) CODE
063213  %       IS VIA SYMBOL-2-LIST.
063213  %
063213  %*************************************************************************
063213
063213  *XXRPT=177777 % Set flag indicating that XMSG has been generated (8XMSG)
063213
063213  %=========================================================================
063213  % 35.37       X 1 I O N   X 1 R E T   X W T 0 5   X G W R D
063213  %
063213  SUBR XWT05
063213
063213  % LEVEL 5 WAIT STATE. NOTE THAT THE PID BIT MUST BE RESET BEFORE CALLING WITH
063213  % INTERRUPTS OFF.
063213  INTEGER POINTER X5HOM
063214
063214  XWT05: A:=L=:"X5HOM"; *PION; TRA STS; PON          % WAIT; GET STATUS; PAGING ON
063221         IF A NBIT 16 THEN                           % CHECK IF PAGING OFF
063223               *TRA PID; COPY SA DD; TRA PIE         % INTERRUPT SYSTEM STATUS
063226               A/\D; CALL TWT04                      % LET TNY HANDLE IT
063230         FI; *IOF
063231         GO X5HOM
063232  RBUS
063233
063233
063233  %=========================================================================
063233  % 35.38       X R T E N
063233  %
063233  %       X R T E N   -   R E S T A R T   R T   F R O M   R T  W A I T
063233  %
063233  %=========================================================================
063233  SUBR XRTEN,XRTDS
063233  % THIS ROUTINE IS CALLED ON LEVEL 5 TO DO AN RTENTRY ON RT-PROG IN A-REG
063233  INTEGER ARRAY XDFLD:=(0,0,XDFLD,0,0,0,XXENT)      % DATAFIELD!
063242  XRTDS: INTEGER ARRAY POINTER RTDS; INTEGER POINTER HOME;
063244
063244  XRTEN: T:=L=:"HOME"; X:=0; *POF                   % A=RT-DESC ADDRESS
063250         DO WHILE 0><RTDS(X); IF A=T GO OUT; X+1; OD % FIND END; CHECK NOT THERE
063257         A=:RTDS(X); X+1; 0=:RTDS(X)                % INSERT IN TABLE
063262         IF X=1 THEN X:="XDFLD":=:B; CALL RTACT; X:=:B; FI % PUT IN MONITOR QUEUE
063271  OUT:   *PON
063272         GO HOME
063273
063273  % MFUNC ROUTINE EXECUTED ON LEVEL 3
063273  XXENT: X:=0; T:=0; *PIOF                          % LOCK OUT LEVEL 5
063276         DO WHILE RTDS(X)><0; X+1; A=:T; OD          % FIND LAST ELEMENT
063303         IF T><0                                    % WAS THERE ANY
063303         THEN X-1; 0=:RTDS(X); *PION                % CLEAR HIS ENTRY
063310              X:=T; CALL RTENTRY; GO XXENT          % START HIM, AND TRY NEXT ONE
```

```
063313          FI; *PION
063314          GO STUPR
063315    RBUS
063321    *"8XMSG+8CXHD+8C1X2+8C2X2+8C3X2+8C4X2
"063321
063321    %
063321    %-------------------------------------------------------------------------------
063321    %
063321    % 35.39   D A T A    C O P Y I N G:   X M S G / P I T 3,  H D L C,  X 2 1
063321    %
063321    %-------------------------------------------------------------------------------
063321    %
063321    SUBR Z2PHY,Z0PHY,Z2USR,Z0USR
063321
063321    % Routines to transfer data between a user V.S. and physical memory. They
063321    % assume that everything has been set up so that only the 0 bit in the status
063321    % register needs to be turned on to allow access via the APT to the user data
063321    % area.
063321
063321    % On entry: D contains the byte count and B points to the parameter block:
063321    DISP 0
063321          INTEGER ZXUBF           % Buffer address in user's logical space
063321          INTEGER ZXSBK,ZXSBF     % Physical buffer address (Bank, Displacement)
063321          DOUBLE ZXSAD=ZXSBK
063321          INTEGER POINTER ZXHOM   % Scratch location used to save L-register
063321    PSID
063321    *APTON=BSET ONE; APTOF=BSET ZRO  % Enable/disable alternative page table
063321
063321    Z0PHY: K:="0"; * APTOF; PON                    % Entry from POF? Set PON and flag
063324          GO ZPHY
063325    Z2PHY: K:="1"                                  % Entry from PON
063326    ZPHY:  D SHZ -1                                      % Convert to word count
063327          IF M                                           % Odd byte count?
063327          THEN X:=ZXUBF+D; *APTON; LDA 0,X; APTOF        % Get via APT
063336               A/\177400                                 % Maskout r.h. byte
063337          FI; T:=ZXSBK; X:=ZXSBF; A:=:L=:"ZXHOM"         % Get physical addr
063343          IF M                                      % Odd count?
063343          THEN X+D; *LDATX 0                             % Get last word
063347               A/\377\/L; *STATX 0                       % Mask in lh byte and store
063352               X-D
063353          FI; ZXUBF:=:D; *APTON                          % TX=Destn addr; D=Useraddr
063356    @LIB CXCPU
063356          X:=:T                                          % XT=Destn address
063357          DO A=:L:=4000:=:L; WHILE A>0                   % A=No of words left,L=4000
063364               *SKP IF DA GRE SL; COPY SA DL             % Skip if more than 4000?
063366               A-L; * MOVAP;                             % A=No of words left
063370          OD;
063371    @ELIB
063371    @LIB CXCPU-,
063371          GO OUT
063372
063372    Z0USR: K:="0"; * APTOF; PON                    % Entry from POF. Set PON & flag
063375          GO ZUSR
063376    Z2USR: K:="1"                                  % Entry with PON
063377    ZUSR:  D SHZ -1;                                     % Convert byte count to words
063400          IF M                                           % Odd byte count?
063400          THEN X:=ZXSBF+D; T:=ZXSBK; *LDATX 0            % Get last word in physbuff
063406               A SHZ -10; D SHZ 1=:X; T:=ZXUBF;          % Mask & get user buffer addr
063412               *APTON; SBYT; APTOF                       % Enable APT & store
063415               GO ZUSR                                   % Try again (even byte count)
```

```
063416            FI;
063416            D=:X; T:=L=:"ZXHOM":=ZXUBF; ZXSAD; *APTON % AD=Physical, X=count, T=user
063424    @LIB CXCPU
063424            DO X=:L:=4000:=:L; WHILE X>0               % X=No of words left,L=4000
063431                  *SKP IF DX GRE SL; COPY SX DL        % Skip if more than 4000?
063433                  X-L; * MOVPA;                        % X=No of words left
063435            OD;
063436    @ELIB
063436    @LIB CXCPU-,
063436    OUT:    * APTOF; BSKP SSK ONE; POF                 % Switch off access to APT
063441            GO ZXHOM
063442    *!KILL APTON APTOF
063442    RBUS
063445    %=============================================================================
063445    %
063445    % 35.40  X M S G / H D L C    I N T E R F A C E  :  S T A R T    D R I V E R
063445    %
063445    %=============================================================================
063445    SUBR ZXS12,ZXS13
063445    INTEGER POINTER HOME
063446    ZXS12: A:=L=:"HOME"; *PIOF                         % X->DCB, B->HDLC DATAFIELD
063451           CALL ICHAIN; CALL ACT12; GO UT              % CHAIN TO HDLC AND KICK HIM
063454
063454    ZXS13: A:=L=:"HOME"; *PIOF                         % X->DCB, B->HDLC DATAFIELD
063457           CALL ICHAIN; CALL ACT13;                    % CHAIN TO HDLC AND KICK HIM
063461
063461    UT:    *PION;
063462           GO HOME
063463    RBUS
063466    *"-8XMSG
"063466
063466    *"8OCTO
"063466     %=============================================================================
063466    %
063466    % 35.42  MONITOR LEVEL ENTRY FOR MON OCTO
063466    %
063466    %=============================================================================
063466    SUBR MOCTBU
063466    MOCTBU:
063466    * PIOF
063467    * IRR ALEVB DT
063470    IF X:=BACKGROUND >< 0 THEN 005; GO ERR FI
063474    IF A >< 0 THEN CALL LOGPH FI
063476    IF A = 0 THEN 240; GO ERR FI
063501    A:=:B
063502    IF RTRES >< X:=RTREF THEN 5; GO ERR FI
063510    BOXNO; * IRW LV13B DT
063512    * IRR ALEVB DA; IRW LV13B DA
063514    * IRR ALEVB DD; IRW LV13B DD
063516    MAINF:    *       IRW LV13B DB
063520    A."STDRIV";*      IRW LV13B DP
063523    L/13;   * MST PID; POF; ION
063527    FINE:     * PION
063530    GO FAR STUPR
063531    ERR:    * IRW ALEVB DA
063532    GO FINE
063533    RBUS
063541
063541    %-----------------------------------------------------------------------%
063541    %-!     O C T 1 3
```

```
063541   %=================================================================%
063541   SUBR OCT13
063541   OCT13: * POF; JPL I (OCTOB
063543   %       GO FAR OCTOBUS
063543   RBUS
063544
063544   %-----------------------------------------------------------------%
063544   % 35.42  A C T O C T                                              %
063544   %        Routine to start octobus driver from direct task level   %
063544   %-----------------------------------------------------------------%
063544   SUBR ACTOCT
063544   SYMBOL PONI  =    160
063544   % t-reg = SLOT_NO        - task number
063544   % a-reg = OFNC           - function
063544   % d-reg = CPU            - destination CPU
063544   %        = P_REG         - start of code to be executed on driver level (13b)
063544   % x-reg = CUR_UNIT       - unit number for actual octobus (0-2)
063544
063544   * IOF
063545   * IRW LV13B DA; COPY ST DA; IRW LV13B DT
063550   * COPY SD DA; IRW LV13B DD; LDA I ,X (OCTOP; IRW LV13B DB
063554   * LDA * 2; JMP * 2; OCT13; IRW LV13B DP
063560   * TRA STS; SAT 1; BSKP ONE PONI DA; SAT 0; COPY ST DD
063565   * SAA 000; BSET ONE LV13B DA; MST PID
063570   * PION
063571   * POF; SKP IF SD EQL 0; PON
063574   *      SKP IF SA EQL 0; EXIT; EXIT AD1
063577   *)FILL
063600   RBUS %                                                  ACTOCT   %
063600   *"
"063600
063600   *"-8N500
"063600    *98END=*
063600
063600
063600   @EOF
063600
```

```
172517
172517
172517
172617
172517   %%%%%%%%%%%%%%%%%%%%%% E X - M R E S - S I N A %%%%%%%%%%%%%%%%%%%%%%%%%%%%
172517
172517   *$8END/
063600
063600   %=============================================================================
063600   %  36.1       G E T 0   G E T 1   G E T 2   G E T 3   G E T 4
063600   %             G E T 5   G E T S 2
063600
063600   % SUBROUTINES TO GET PARAMETER VALUES FROM USER
063600   % ENTRY:      B=PARAMETER LIST
063600   %             X=WORKING AREA
063600   % RETURN:     X=B=WORKING AREA
063600   % RT LEVEL
063600
063600   SUBR GET0,GET1,GET2,GET3,GET4,GET5,GET6,GETS2
063600   DISP 0; DOUBLE POINTER DP1=P1; DOUBLE DD1=D1; PSID
063600   GETS2: DP1; *BSET ZRO 0; STD DD1,X; BSET ONE 0; JMP GET1
063605   GET6:  P5; *BSET ZRO 0; STA D5,X; BSET ONE 0
063611   GET5:  P4; *BSET ZRO 0; STA D4,X; BSET ONE 0
063615   GET4:  P3; *BSET ZRO 0; STA D3,X; BSET ONE 0
063621   GET3:  P2; *BSET ZRO 0; STA D2,X; BSET ONE 0
063625   GET2:  P1; *BSET ZRO 0; STA D1,X; BSET ONE 0
063631   GET1:  P0; *BSET ZRO 0; STA D0,X; BSET ONE 0
063635   GET0:  X=:B; *PIOF
063637          IF BACKGROUND><0 THEN L=:D;*PION
063643               CALL XBMRET; D=:L; *POF
063646          FI
063646          RTREF.ACTPRI/\177177=:X.ACTPRI/\3773; B=:X; *TRR PCR
063655          MLEV; *MCL PIE; TRA PGS; PION
063661          EXIT
063662   RBUS
```

```
063667
063667
063667
063667  % ============================================================================
063667  % 36.2        P M T R A N S
063667
063667  DOUBLE SWPCOREADR,SWPMASSADDR
063673  INTEGER SWPMTMODUS,SWPBLPAGE
063675  INTEGER SWPPARLIST:=(SWPMTMODUS,SWPCOREADR,SWPMASSADR,SWPBLPAGE)
063701
063701  SUBR PMTRANS
063701  INTEGER POINTER LREG
063702  PMTRANS: T:=L=:"LREG"; CALL MTRANS; *POF
063706          GO LREG
063707  RBUS
063707
063707  SUBR PAGEFAULT,PAGE2FAULT,PWFAIL,RESYS,COLDSTART
063707  RBUS
063707
063707  %============================================================================
063707  % 36.3        P P A G E F A U L T   P 2 P A G E 2 F A U L T
063707  %            P P W F A I L   P R E S Y S
063707  %            P C O L D S T A R T
063707
063707  SUBR PPAGEFAULT,P2PAGE2FAULT,PPWFAIL,PRESYS,PCOLDSTART
063707  PPAGEFAULT: *POF
063710          GO PAGEFAULT
063711  P2PAGEFAULT: *POF
063712          GO PAGE2FAULT
063713  PPWFAIL: *PIOF
063714          GO PWFAIL
063715  PRESYS: *PIOF
063716          GO RESYS
063717  PCOLDSTART: *PIOF
063720          GO COLDSTART
063721  RBUS
063726
063726  %============================================================================
063726  % 36.4        A T R A N S      X T R A N S    D B T R A N S
063726  %
063726  % SUBROUTINE TO TRANSFORM ADDRESSES TO WINDOW ADDRESSES
063726  % CALLED FROM BLOCK MONITOR CALLS (MAGTP,RFILE,..) AND XMSG
063726  % RT LEVEL
063726  %
063726  % ATRANS:     PAGING/INTER. STATUS. ON CALL: UNDEFINED, RETURN: PION
063726  % ENTRY:      A=USER ADDRESS,  X=BUFFER ADDRESS, D=CALLING RT-PROGRAMS ACTPRI
063726  % RETURN:     A=WINDOW ADDRESS, X=BUFFER WINDOW
063726  %
063726  % DBTRANS:    PAGING/INTERRUPT STATUS AS FOR ATRANS.
063726  % ENTRY:      AS FOR ATRANS BUT T-REG MUST CONTAIN NO. OF BANK IN WHICH BUFFERS LIE.
063726  % EXIT:       AS FOR ATRANS.
063726
063726  % XTRANS:     PAGING/INTER. STATUS MUST BE PIOF WHEN CALLING XTRANS (AND RETURN)
063726  % ENTRY:      AS ATRANS, BUT T=BANK NO OF BUFFER (0 TO 3)
063726  % RETURN:     AS ATRANS
063726
063726  SUBR ATRANS,XTRANS,DBTRANS
063726  INTEGER TREG,AREG,DREG,XREG; REAL TADREG=TREG
063732  DOUBLE POINTER BPG:=177000+5BFPAGE+5BFPAGE
063733  DOUBLE POINTER WND0:=177174,WND1:=177176
```

```
063735    INTEGER C5BFPAGE(0); *5BFPA@12                    % "5BFPA*2000"
063736
063736    XTRANS: X=:XREG
063737    FELLS:  TAD=:TADREG; 600/\D SHZ -1; T:=AREG SHZ -12\/A % USER LOG PAGE
063746            XREG/\176000=:D:=TREG; AD SHZ -2
063753    @LIB CXCPU-,
063753    @LIB CXCPU
063753            T=:TREG; A:=RTREF.WINDOW/\377\/D=:X.WINDOW
063761            D SHZ -10; A:=162000; AD=:BPG
063764            T:="5UBFPAGE*2000"
063765    @ELIB
063765            1777/\XREG\/C5BFPAGE=:X                   % LOG. ADDR OF SYSTEM BUFFER
063771            1777/\AREG\/T=:AREG                       % LOG. ADDR OF USER BUFFER
063775            TAD:=TADREG; EXIT
063777
063777    INTEGER ATSAV,ALSAV,BANKSAV                       % SAVE LOCATIONS FOR ATRANS AND DBTRANS
064002    DBTRANS:
064002              *PIOF
064003              T=:ATSAV=:BANKSAV                       % BANK IN WHICH DEVICE BUFFERS LIE.
064005    @LIB CXCPU
064005              X=:XREG:=L=:ALSAV; CALL FELLS; T=:L:=ALSAV; *PION
064014              T:=:L; T=:UWLOGADR; EXIT
064017    @ELIB
064017    @LIB CXCPU-,
064017
064017    ATRANS:   *PIOF
064020              T=:ATSAV:=0=:BANKSAV                    % BANK 0
064023    COMM:     X=:XREG:=L=:ALSAV
064026              CALL FELLS; T:=ALSAV=:L:=ATSAV; *PION   % RESTORE REGISTERS
064033            EXIT
064034    RBUS
064043
```

```
064043
064043   %=============================================================================
064043   %
064043   %                  S E G M E N T   M O N I T O R   C A L L S
064043   %
064043   %-============================================================================
064043   % 36.5        M C A L L   M E X I T
064043
064043   % MONITOR CALLS MCALL AND MEXIT - TO CHANGE SEGMENTS
064043   % APPL. LEVEL
064043
064043   SUBR MCALL,MEXIT,OLEGSEG,LEGSEG,R15ERRD,RLEGSEG
064043
064043   DISP 0; DOUBLE MSTADR; PSID
064043   DISP 0; INTEGER MSEGM=D0; PSID
064043   INTEGER STRA,LA,STRB,LB,MSEGA,MSEGB
064051   INTEGER POINTER LREG
064052
064052   MCALL: T=:B; MSTADR; CALL ALTOFF; T:=X.ZPREG=:X.ZLREG
064057          A=:X.ZPREG:=D=:X.MSEGM; CALL GET0; GO MMC
064064
064064   MEXIT: CALL GET0; ZLREG=:ZPREG; T=:MSEGM
064070   MMC:   RTREF.ACTSEG=:ZTREG
064073          IF MSEGM SHZ -10=377 THEN X.ACTSEG SHZ -10 FI
064102          A=:MSEGA; CALL DECO; A=:STRA; T=:LA; X:=RTREF
064107          IF MSEGM/\377=377 THEN X.ACTSEG/\377 FI
064116          A=:MSEGB; CALL DECO; T=:LB
064121   %CHECK FOR OVERLAP:
064121          IF A>STRA THEN IF A-LA<T GO ERRO
064127          ELSE IF A+LB>T GO ERRO
064133          FI
064133          MSEGA SH 10+MSEGB=:RTREF.ACTSEG
064140          GO RETSTUPR
064141   ERRO:  MSEGA SH 10+MSEGB; CALL 9ERRA(#42); GO RETXIT  % OVERLAP
064147
064147   % DECODE SEGMENT ELEMENT
064147   % ENTRY: A=SEGMENT NUMBER; X=RTREF
064147   % EXIT:  A=FIRST LOGICAL PAGE IN SEGMENT; T=NUMBER OF PAGES IN SEGMENT
064147   DECO:  IF A=0 THEN T:=0; EXIT FI
064152          T:=L=:"LREG"; IF A=X.RSEGM GO ERRIL
064157          CALL LEGSEG; *POF
064161          CALL SHRSOVERLAP; *PON
064163          T:=X.LOGADR SHZ -10; X.LOGADR/\377
064167          GO LREG
064170
064170   % SUBROUTINE TO CHECK FOR LEGAL SEGMENT
064170   % ENTRY:      A=SEGMENT NUMBER
064170   % EXIT:       A=SEGMENT FLAG; D=SEGMENT NUMBER; X=SEGMENT TABLE ADDRESS
064170   % RETURN TO RETXIT IF ERROR
064170
064170   OLEGSEG: IF A=0 GO ERRIL
064171   LEGSEG:  IF A>>X:=SGMAX GO ERRIL
064174            A=:D*5SEGSIZE+SEGSTART=:X
064200            IF X.FLAG=0 OR A BIT 5INHB GO ERRNL
064204            EXIT
064205
064205   R15ERRD:
064205   ERRIL: CALL 9ERRA(#15); GO RETXIT                  % ILL.SEGM.NO.
064210   ERRNL: A:=D; CALL 9ERRA(#16); GO RETXIT            % SEGMENT NOT LOADED
064214
```

```
064214    % CHECK FOR LEGAL SEGMENT NUMBER IN MON REENT/SREENT CALLED FROM BACKGROUND
064214    % PROGRAMS. ONLY SEGMENTS ON PIT 1 AND PIT 2 WITH PROTECTION RING ZERO
064214    % IS LEGAL
064214    %
064214    RLEGSEG: IF BACKGROUND><0 AND OLDPAGE/\3=0 THEN
064221                  IF X.LOGADR/\300><100 AND A><200 GO ERRIL
064231                  IF X.FLAG/\3><0 GO ERRIL
064234             FI; EXIT
064235    RBUS
064257
064257    %-------------------------------------------------------------------------
064257    % 36.6        M F I X   M U N F I X
064257
064257    %MONITOR CALLS: CALL FIX(SEGNO),CALL UNFIX(SEGNO)
064257    SUBR MOFIX,MUNFIX
064257    DISP 0; INTEGER SGADR=D1,OLDSGNO=D2; PSID
064257
064257    MOFIX: CALL GET1; D0; CALL OLEGSEG
064262           IF A BIT 5DEMAND GO ERRD
064264           IF A BIT 5FIX GO RET
064266           IF X.LOGADR SHZ -10=0 GO ERRNL
064271           IF A+FIXPAGES>FIXMAX GO ERRF
064275           A=:FIXPAGES; X=:SGADR
064277           RTREF.ACTSEG=:OLDSGNO; D0=:X.ACTSEG
064304           "STUPR"; *IRW MLEVB DP
064306           MLEV; *MST PID; MST PIE
064311    %RETURN WITH CHANGED SEGMENT
064311           MLEV; *MCL PIE
064313           OLDSGNO=:X.ACTSEG    %CHANGE BACK
064315           SGADR.FLAG BONE 5FIX=:X.FLAG
064321           X=:L:="BSEGLINK"
064323    LOOP:  X=:D:=X.SEGLINK; IF X=-1 THEN CALL ERRFATAL FI
064331           IF X><L GO LOOP
064333           X.SEGLINK; 0=:X.SEGLINK; A=:D.SEGLINK
064337           GO RETSTUPR
064340
064340    MUNFIX: CALL GET1; IF D0=0 OR A>>SGMAX GO ERRIL
064346           A*5SEGSIZE+SEGSTART
064350           IF A.FLAG BIT 5FIX THEN
064354                 A BZERO 5FIX=:X.FLAG
064356                 377; T:=X.LOGADR SHZ -10/\A; FIXPAGES-T=:FIXPAGES; IF <0 GO ERRF
064366                 IF X.SEGLINK><0 THEN CALL ERRFATAL FI
064371                 BSEGLINK=:X.SEGLINK; X=:BSEGLINK
064374           FI; GO RET
064375
064375    ERRNL: D0; CALL 9ERRA(#16); GO RETXIT    % EMPTY SEGMENT (NOT LOADED)
064401    ERRD:  D0; CALL 9ERRA(#17); GO RETXIT    % FIXING DEMAND
064405    ERRF:  D0; CALL 9ERRA(#18); GO RETXIT    % TOO MANY FIXED
064411    ERRIL: D0; CALL 9ERRA(#15); GO RETXIT    % ILLEGAL SEGMENT NUMBER
064415    RBUS
064435
064435    %-------------------------------------------------------------------------
064435    % 36.7        W S E G   -   W S E G X
064435
064435    % MONITOR CALL: CALL WSEG(SEGNO) - WRITE BACK SEGMENT
064435    SUBR WSEG,WSEGX
064435
064435    INTEGER SG,SGA
064437    INTEGER POINTER CRETADR
064446
```

```
064440   WSEG:  CALL GET1
064441          "RET"=:"CRETADR"
064443          D0=:SG; CALL OLEGSEG; X=:SGA
064447   FELLS: *POF
064450          CALL SRESER; *PON
064452          "MWSEG"; *IRW MLEVB DP
064454          MLEV; *MST PID; MST PIE
064457
064457   %MONITOR LEVEL:
064457   MWSEG: X:=RTREF=:SRTREF
064461          IF SGA.FLAG=0 OR A BIT 5INHBT THEN
064466             SG; CALL 9ERR(#16); GO BRTEXT
064472          FI; A BZERO 50K=:X.FLAG; X=:B=:SEGREF:=SRTREF
064477          *POF
064500          A:=B; CALL CSEGS; 1=:NUMBER; CALL TRNSEG
064505          X:=SRTREF; CALL ZSRELES; "OUT"; *PON
064511          IF X=CURPROG THEN
064514             *IRW ALEVB DP
064515          ELSE
064516             X:=X.RTDLGADDR; T:=0; *STATX XTDPR
064521          FI; GO STUPR
064522   %RT LEVEL:
064522   OUT:   GO CRETADR
064523
064523   % CALLED FROM PSPLREE IN POF
064523   % ENTRY:      A=SEGMENT NUMBER
064523   %            X=ADDRESS OF SEGMENT TABLE ENTRY
064523   %
064523
064523   INTEGER POINTER WSEGL
064524   WSEGX: A=:SG:=L=:"WSEGL":="WSEG1"=:"CRETADR"; X=:SGA; GO FELLS
064533   WSEG1: GO WSEGL
064534   RBUS
064556
064556   %=============================================================================
064556   % 36.8          R E E N T
064556
064556   % MONITOR CALL REENT(SEGNO) - ATTACH REENTRANT SEGMENT
064556   SUBR REENT,IMREENT
064556   DISP 0; INTEGER RESEG=D1; PSID
064556
064556   REENT: CALL GET1; *POF
064560   IMREENT:  CALL XCSEGS
064561          IF D0><0 THEN
064563             CALL LEGSEG
064564             X=:RESEG; IF A NBIT 5DEMAND GO ERRD
064567             IF X=NSEGA OR X=NSEGB GO RETSTUPR
064575             CALL RLEGSEG
064576             T:=D0; NSEGA.FLAG; CALL SHRSOVERLAP
064602             T:=D0; NSEGB.FLAG; CALL SHRSOVERLAP
064606             NSEGA; T:=RESEG; CALL OVERLAP
064611             NSEGB; T:=RESEG; CALL OVERLAP
064614          FI
064614          X:=RTREF; CALL CLRBMAP
064616   OUT:   RTREF.RSEGM/\177400+D0=:X.RSEGM; GO RETSTUPR
064624
064624   ERRD:  CALL 9ERRA(#45); GO RETXIT  % NOT DEMAND SEGMENTS
064627   RBUS
064645
064645
```

```
064645   %-------------------------------------------------------------------------
064645   %              S P L R E E
064645   % MON SPLREE(SGNO,FP1,NP1,FP2,LP2,FLAG)
064645   %              FP1= FIRST PAGE IN LOG.AREA 1 TO OVERLAP BACKGROUND SEGMENT
064645   %              NP1= NUMBER OF PAGES IN AREA 1
064645   %              FP2= FIRST PAGE IN LOG.AREA 2 TO OVERLAP BACKGROUND SEGMENT
064645   %              NP2= NUMBER OF PAGES IN AREA 2
064645   %              FLAG><0: CLEAR PAGE-OWNER TABLE
064645   %
064645   SUBR SPLRE,3SPLRE
064645   SPLRE: CALL GET6; *POF
064647          CALL XCSEGS; CALL S3NOALTPIT; *PON
064652          GO 3SPLRE
064653   RBUS
064657
064657
064657   %=========================================================================
064657   % 36.10       C H C O R M A P
064657   %
064657   SUBR CHCORMAP
064657   CHCORMAP: EXITA
064660   RBUS
064660
064660   %-------------------------------------------------------------------------
064660   %              S S S W P   -   C B R E L   -   C B R E S
064660   %
064660   % RT-PROGRAM RESERVING THE SWAPPING-SEMAPHORE (CLFIE) AND WAITING FOR
064660   % THE DISC ON BEHALF OF THE SWAPPING-REQUESTING PROGRAM
064660   % THE SWAPPING-REQUESTING PROGRAM WILL BE SET IN SWAPPING-WAIT
064660   % 1SWAP HAS THE PRIORITY OF THE SWAPPING-REQUESTING PROGRAM
064660   %
064660   SUBR INRWSEGM,SSSWP,CBREL,CBRES,LNK1SWAP
064660   INTEGER XREGG
064661   SSSWP: *IOF
064662          "XRTEX"; *IRW MLEVB DP
064664          MLEV; *MST PID; ION
064667          GO SSSWP
064670
064670   % MONITOR LEVEL
064670   XRTEX: X:="1SWAP"
064671          DO WHILE X.BRESLINK><X; A=:B; CALL BRELEASE; OD
064677          CALL FREXQ
064700          IF SWPFLAG><1 THEN
064704             X:="BEXQU-WLINK"; T:="BEXQU-BWLINK"; A:=XREGG
064707             DO WHILE X:=X.WLINK><T
064712                IF X=A GO IRWAIT
064714             OD
064715          FI; GO STUPR
064716
064716   % SUBROUTINES FOR QUICK BRESERVE/BRELEASE
064716   CBREL: X+5BRESLINK
064717          DO WHILE X.RESLINK><B; X:=A; OD
064724          RESLINK=:X.RESLINK; 0=:RESLINK=:RTRES; EXIT
064731
064731   CBRES: X.BRESLINK=:RESLINK; A:=B=:X.BRESLINK; X=:RTRES; EXIT
064737
064737   %-------------------------------------------------------------------------
064737   %              L N K 1 S W A P
064737   %
064737   % ROUTINE TO START 1SWAP
```

```
064737    % MONITOR LEVEL, CALLED FROM RWSEGM
064737    %
064737    % ENTRY:      B=ADDRESS OF DISC DATAFIELD
064737    %             X=DISC-REQUESTING RT-PROGRAM
064737    %
064737    % EXIT TO PSTUPR
064737    %
064737
064737    1NK1SWAP: X=:XREGG
064740            X.ACTPRI BONE 5SWWAIT=:X.ACTPRI
064743            X.STATUS BONE 5WAIT=:X.STATUS
064746            X:="1SWAP"; CALL TOWQU                    % 1SWAP WAITS FOR DISC
064750            X:=XREGG; "CLFIE"=:B; CALL CBREL          % RELEASE SWAPPING-SEMAPHORE FROM
064754                                                      % DISC-REQUESTING PROGRAM
064754            X:="1SWAP"; CALL CBRES                    % DISC IS RESERVED BY 1SWAP
064756            GO PSTUPR
064757     RBUS
064776
064776    %=============================================================================
064776    %             S W P R E L E A S E
064776    %
064776    % RESETT THE SWAPPING-QUEUE
064776    %
064776    % ENTRY:      B=DATAFIELD
064776    %
064776    SUBR SWPRELEASE
064776    INTEGER XREGG
064777    SWPRELEASE:
064777            IF "CLFIE"=B THEN
065002               X=:XREGG:="BEXQUE-WLINK"; T:="BEXQU-BWLINK"
065005               DO WHILE X:=X.WLINK><T
065010                  IF X.ACTPRI BIT 5SWWAIT THEN
065013                     A BZERO 5SWWAIT=:X.ACTPRI; X.STATUS BZERO 5WAIT=:X.STATUS
065020                  FI
065020               OD; X:=XREGG
065022            FI; EXIT
065023    RBUS
065026
065026    %=============================================================================
065026    %             M R L C L F I E
065026    %
065026    % SUBROUTINE TO RELEASE THE SWAPPING-SEMAPHORE (CLFIE) WHEN
065026    % WAITING FOR THE DISC (WAITING IN RWSEGM)
065026    %
065026    % CALLED FROM MESCAPE AND SYSABORT IN MON-60 (ND-500 DRIVER)
065026    %
065026    % ENTRY: X=ACTUAL PROGRAM
065026    %
065026    SUBR MRLCLFIE
065026    MRLCLFIE: IF SWPFLAG><0 THEN
065030                  IF X.ACTPRI BIT 5SWWAIT THEN
065033                     A BZERO 5SWWAIT=:X.ACTPRI
065035                     X.STATUS BZERO 5WAIT=:X.STATUS
065040                  FI
065040               FI; EXIT
065041    RBUS
065042
065042    *'8DIL%
"06504,"
065042    %=============================================================================
```

```
065042    %              M L D I L
065042    %
065042    % MONITOR LEVEL ROUTINE TO START OR STOP THE DISC ACCESS LOG RT-PROGRAM
065042    %
065042    SUBR MLDIL
065042    MLDIL: "RTDIL".STATUS BZERO SWAIT=:X.STATUS; CALL RTENTRY
065047           GO MONEN
065050    RBUS
065053
065053    *"
"065053
065053    %=============================================================================
065053    % 36.13       E N T S G
065053    %
065053    %  MONITOR CALL: ENTSG(SEGNO,PT.NO,INT.LEVEL,STARTADDR)
065053    %  ENTERING A FIXED SEGMENT INTO THE PAGE TABLES
065053    SUBR ENTSG
065053    INTEGER SVB,IRWINST(0); *IRW DP
065055    ENTSG: CALL GET4
065056           IF D1>>3 GO ERR                        % ILLEGAL PAGE TABLE NUMBER
065062           IF D0>>SGMAX GO ERR
065066           A*5SEGSIZE+SEGSTART
065070           IF A.FLAG NBIT 5FIX GO ERR
065074           D1; *PIOF
065076           A SH 7+177000:=:B=:SVB; T:=CORMBANK; X:=X.BPAGLINK
065104           DO WHILE X><0
065105              *LDATX DALOG
065106              A/\77 SH 1+B=:B; *LDDTX DPGPR
065113              AD=:D0U0; A:=B/\177600=:B; *LDXTX DPAGL
065120           OD; SVB=:B; *PION
065124           IF D2<2 OR >11 OR =3 OR =4 GO ERR
065141           D1 SH 7=:D SH 2; D\/A; D2 SH 3+2\/D; *TRR PCR
065153           D2 SH 3\/IRWINST=:T; D3; *EXR ST
065161           1; FOR X:=D2- DO A SH 1; OD; *MST PIE
065167           0=:ZAREG; GO RET
065171    ERR:   -1=:ZAREG; GO RET
065174    RBUS
065205
065205    %=============================================================================
065205    %              F X C T A
065205    %
065205    % TABLE FOR START OF COREMAP FOR ALLOCATED MEMORY AREAS (NOT SEGMENTS)
065205    % LAYOUT: WORD 1: MEMORY MAP POINTER OF THE ALLOCATED MEMORY AREA
065205    %         WORD 2: PROGRAM WHICH HAS ALLOCATED THE MEMORY AREA
065205
065205    INTEGER ARRAY FXCTAB(0)
065205    * FXCTA<*+NALME+NALME; )ZERO; *+NALME+NALME/
065217
065217
065217    INTEGER PAGPN                  % MEMORY MAP ELEMENT ADDR OF FIRST PAGE
065220                                   % LINKED OUT BY MON 61
065220
065220    %-----------------------------------------------------------------------------
065220    % 36.14       F I X C   P F I X C   F I X C 5 0 0
065220    %
065220
065220    SUBR FIXC,PFIXC,FIXC500
065220
065220    BASE BFIXC
065220    INTEGER SGADR,FPHYS,SG1ADR:=XSGRT,OLDFLAG
```

```
065224    INTEGER CMMENTRY,CSEGN,STEPINCR,CCORMSTART,CECORMAP,BREG
065232    INTEGER COLDFLAG,CINDX
065234    INTEGER PGFOUND,PREVPGFOUND
065236    INTEGER CFPHYS,CT,CA,CD,CLOGPAGE,CCECORMAP
065244    INTEGER POINTER LINK
065245    TRIPLE CTAD=CT
065245    INTEGER FIXFLAG
065246    INTEGER NPAGES; INTEGER CCTINDX=NPAGES
065247    INTEGER AFPHYS,NPARTS=AFPHYS
065250    INTEGER ALPHYS; DOUBLE ARRAY POINTER DPARTADDR=ALPHYS
065251    INTEGER NSHARED; INTEGER POINTER RNPARTS=NSHARED; INTEGER MXMININTERVAL=NSHARED
065252    INTEGER ARRAY POINTER ARRPNT; DOUBLE ARRAY POINTER DRETADDR=ARRPNT
065253    DOUBLE ARRAY POINTER DARRPNT=ARRPNT
065253    TRIPLE ARRAY POINTER FARRPNT=ARRPNT
065253    INTEGER ARRAY POINTER PCCTAB:=CCTAB
065254    DOUBLE ARRAY POINTER DPCCTAB=PCCTAB
065254    DOUBLE POINTER DCCFPA:=CCFPA
065255    INTEGER PREVT,PREVX
065257    TRIPLE T5D1=NPAGES; DOUBLE D5DD4=NSHARED
065257    INTEGER CURPAGE,CPEND; DOUBLE CCAREA=CURPAGE
065261    INTEGER CNPAGES=CPEND, CSGNO=CPEND, CTINDX=CPEND
065261    INTEGER CBPGLINK
065262    INTEGER PARTINDX,RETINDX
065264    INTEGER POINTER RINDXADDR
065265    INTEGER SVFLAGB
065266    ESAB
065266
065266    DISP 0; TRIPLE TD1=D1; DOUBLE DD4=D4; PSID
065266    INTEGER POINTER PPTTIF:=TTIFIELD
065267
065267    FIXC:   CALL GET2; D1=:"BFIXC".FPHYS
065273            IF DO BZERO 17<=0 OR A>SGMAX GO ERRIL      % ILLEGAL SEGMENT NUMBER
065302            A*5SEGSIZE+SEGSTART=:X.SGADR
065305            IF A.FLAG=0 OR A BIT 5INHB GO ERRNL        % SEGMENT NOT LOADED
065312            IF A BIT 5DEMAND GO ERRD                   % DEMAND SEGMENT NOT LEGAL
065314            T:="BFIXC"; T:=:B; T=:BREG
065317            IF A BIT 5FIX THEN                         % SEGMENT ALREADY FIXED
065321    % CHECK IF FIXED CONTINUOUS IN CORRECT ADDRESS
065321              FPHYS=:CURPAGE; X:=X.BPAGLINK; T:=CORMBANK
065325              DO
065325                 *LDATX DPAGP
065326                 IF A-CURPAGE><0 THEN BREG=:B; GO ERRFX FI  % ERROR, FIXED IN WRONG ADDR.
065333                 MIN CURPAGE; *LDXTX DPAGL
065335              WHILE X><0
065336              OD; BREG=:B
065341              IF DO BIT 17 THEN A:=0; GO FXRET FI; GO RETSTUPR % SEGMENT FIXED OK.
065347            FI
065347            IF SGADR.LOGADR SHZ -10=0 THEN BREG=:B; GO ERRNL FI % ERROR, EMPTY SEGMENT
065356            A=:NPAGES
065357            IF A+FIXPAGES>FIXMAX THEN BREG=:B; GO ERRF FI  % TOO MANY FIXED PAGES
065366            0=:FIXFLAG
065367    C1:     IF BACKGROUND><0 THEN
065371               PPTTIF.FLAGB=:SVFLAGB; CALL ESCOFF
065375            FI
065375            CALL S3NOALPIT                    % ALT PIT=NORM PIT=3
065376            GO PFIXC; *)FILL
065414
065414
065414    ERRIL:  IF DO BIT 17 THEN A:=-2; GO FXRET FI
065421            CALL 9ERRA(#15); GO RETXIT         % ILL.SEGM.NO.
```

```
065424   ERRNL: IF DO BIT 17 THEN A:=-3; GO FXRET FI
065431          CALL 9ERRA(#16); GO RETXIT     % NOT LOADED
065434   ERRD:  IF DO BIT 17 THEN A:=-4; GO FXRET FI
065441          CALL 9ERRA(#17); GO RETXIT     % FIXING DEMAND
065444   ERRF:  IF DO BIT 17 THEN A:=-5; GO FXRET FI
065451          CALL 9ERRA(#18); GO RETXIT     % TOO MANY PAGES FIXED
065454   ERRFX: IF DO BIT 17 THEN A:=-6; GO FXRET FI
065461          CALL 9ERRA(#25); GO RETXIT     % ALREADY FIXED
065464
065464   FXRET: A=:ZAREG; GO RETSTUPR           % RETURN TO USER PROGRAM
065466
065466   % ================================================================
065466   % SPECIAL MONITOR CALL TO ALLOCATE MEMORY SPACE FOR NORD-500 SEGMENTS
065466   % AND/OR FIXC (IN MEMORY AREA)/ ALLOCATE BUFFER (IN MEMORY AREA)
065466   %
065466   %
065466   % FUNCTION=1:       ALLOCATE MEMORY FOR NORD 500 SEGMENT - ONE USER VERSION
065466   %
065466   % PAR1:        NUMBER OF PAGES IN SEGMENT
065466   % PAR2:        FIRST LEGAL PHYSICAL PAGE FOR SEGMENT
065466   % PAR3:        LAST LEGAL PHYSICAL PAGE FOR SEGMENT
065466   % PAR4:        NUMBER OF SINTRAN III/N500 SHARED AREAS
065466   % PAR5:        ADDRESS OF SHARED INFORMATION ARRAY (PHYSICAL ADDRESS)
065466   %
065466   % RETURN:      ERROR - NO AREA ALLOCATED FOR NORD-500
065466   %
065466   % SKIP RETURN:  MEMORY ALLOCATED FOR NORD-500 SEGMENT
065466   %                A=FIRST PHYSICAL PAGE IN NORD-500 SEGMENT
065466   %                NUMBER OF FIXED SINTRAN III SEGMENTS  IS RETUNED
065466   %                IN THE "SHARED INFORMATION ARRAY"
065466   %
065466   %
065466   % FUNCTION=2:     GIVE N500 SWAPPING PAGES
065466   %
065466   % PAR1:        NO. OF PAGES
065466   % PAR2:        NO. OF N500 HARDWARE CONFIGURATION MEMORY PARTS
065466   % PAR3:        ADDRESS OF N500 HARDWARE CONFIGURATION MEMORY PARTS TABLE
065466   %              (PHYSICAL ADDRESS)
065466   % PAR4:        MAX NO. OF MEMORY INTERVALS TO RETURN
065466   % PAR5:        ADDRESS OF TABLE TO RETURN THE AVAILABLE MEMORY PARTS INTO
065466   %              (PHYSICAL ADDRESS)
065466   %
065466   % RETURN:      ERROR
065466   % SKIP RETURN: A = NO. OF MEMORY INTERVALS
065466   %
065466   %
065466   % FUNCTION=3:    TAKE PAGES FROM N500
065466   %
065466   % PAR1:        NO. OF PAGES
065466   % PAR2:        NO. OF MEMORY PARTS
065466   % PAR3:        ADDRESS OF MEMORY PARTS TABLE (PHYSICAL ADDRESS)
065466   % PAR4:        DUMMY
065466   % PAR5:        DUMMY
065466   %
065466   % RETURN:      ERROR
065466   % SKIP RETURN: OK
065466   %
065466   %
065466   % FUNCTION=4:    FIX A SEGMENT CONTIGOUSLY AT ANY ADDRESS WITHIN A
065466   %                SPECIFIED AREA
```

```
065466   %
065466   % PAR1:       SEGMENT NUMBER
065466   % PAR2:       FIRST (LOWER) LEGAL PHYSICAL PAGE NUMBER
065466   % PAR3:       LAST (UPPER) LEGAL PHYSICAL PAGE NUMBER
065466   % PAR4:       DUMMY
065466   % PAR5:       DUMMY
065466   %
065466   % RETURN:     A=0: AREA OCCUPIED
065466   %             A=1: "PAR2">>LAST PAGE IN MEMORY MAP
065466   %             A=2: SOMETHING WRONG WITH THE SEGMENT (DEMAND, ALREADY FIXED...)
065466   %
065466   % SKIP RETURN:  SEGMENT IS FIXED CONTIGOUSLY, A=FIRST PHYSICAL PAGE IN SEGMENT
065466   %
065466   %
065466   % FUNCTION=5:   RESERVE A CONTIGOUS MEMORY AREA
065466   %
065466   % PAR1:       NUMBER OF PAGES TO RESERVE
065466   % PAR2:       FIRST (LOWER) LEGAL PHYSICAL PAGE
065466   % PAR3:       LAST (UPPER) LEGAL PHYSICAL PAGE
065466   % PAR4:       DUMMY
065466   % PAR5:       DUMMY
065466   %
065466   % RETURN:     ERROR:
065466   %             A=0: AREA OCCUPIED
065466   %             A=1: "PAR2" >> LAST PAGE IN MEMORY MAP
065466   %             A=2: NO FREE TABLE ELEMENT (TRYING TO RESERVE TO MANY AREAS)
065466   %
065466   % SKIP RETURN:  OK, A=FIRST PHYSICAL PAGE IN AREA, T=TABLE INDEX
065466   %
065466   %
065466   % FUNCTION=6:   RELEASE MEMORY AREA RESERVED BY FUNCTION=5
065466   %
065466   % PAR1:       TABLE INDEX (RETURN PARAMETER FROM FUNCTION=5)
065466   % PAR2:       FIRST PHYSICAL PAGE IN AREA (FROM FUNCTION=5)
065466   % PAR3:       DUMMY
065466   % PAR4:       DUMMY
065466   % PAR5:       DUMMY
065466   %
065466   % RETURN:     ERROR
065466   %
065466   % SKIP RETURN:  OK
065466   %
065466   %
065466   % FUNCTION=7:   RELEASE ALL MEMORY AREAS RESERVED BY A SPECIFIC PROGRAM.
065466   %               THE MEMORY AREAS MUST BE RESERVED BY FUNCTION=5 IN MON 61.
065466   %
065466   % PAR1:       RT-PROGRAM WHICH HAS RESERVED THE MEMORY AREAS
065466   %             PAR1=0 MEANS CALLING PROGRAM.
065466   % PAR2:       DUMMY
065466   % PAR3:       DUMMY
065466   % PAR4:       DUMMY
065466   % PAR5:       DUMMY
065466   %
065466   % RETURN:     ERROR
065466   %
065466   % SKIP RETURN: OK
065466   %
065466   %
065466   %
065466   % FUNCTION=10:  GET SEGMENT'S FIXED-STATUS
```

```
065466    %
065466    % PAR1:       SEGMENT NUMBER
065466    % PAR2:       PAGE NUMBER WITHIN SEGMENT TO FIND THE PHYSICAL MEMORY PAGE OF,
065466    %             IF THE SEGMENT IS FIXED.
065466    %             <PAR2>=0 IS ALLWAYS FIRST PAGE IN SEGMENT.
065466    % PAR3:       DUMMY
065466    % PAR4:       DUMMY
065466    % PAR5:       DUMMY
065466    %
065466    % RETURN:     ERROR; A=174 : ILLEGAL SEGMENT NUMBER <PAR1> OR ILLEGAL PAGE
065466    %                            NUMBER WITHIN SEGMENT <PAR2>
065466    %                    A=2    : SEGMENT NOT LOADED
065466    %
065466    % SKIP RETURN: OK, T=FIXED STATUS
065466    %                   T=0 : SEGMENT NOT FIXED
065466    %                   T=1 : SEGMENT FIXED SCATTERED
065466    %                   T=2 : SEGMENT FIXED CONTIGUOUSLY
065466    %                   A=PHYSICAL PAGE NUMBER WHER <PAR2> PAGE RESIDE IN MEMORY
065466    %                   D=SEGMENT-TABLE-ELEMENT.FLAG
065466    %
065466
065466    FIXC500:
065466            CALL GET6
065467            IF D0>>10 GO ERET; A=:"BFIXC".FIXFLAG
065475            IF A<4 AND PN500=0 GO ERET
065502            TAD:=TD1=:X.T5D1; AD:=DD4=:X.D5DD4
065506            X:=:B=:BREG; GO L1
065511    ERET:   174=:ZAREG; GO RET
065514    RBUS
065523
065523    %=============================================================================
065523    % 36.15       S T M L E V   Y T R N S E G   X T R N S E G
065523    %
065523    SUBR STMLEV,YTRNSEG,XTRNSEG
065523    STMLEV: *PON; JMP *+1; ION; JMP *+1; POF; EXIT
065531    YTRNSEG: *POF
065532            GO XTRNSEG
065533    RBUS
065534
065534
065534    %=============================================================================
065534    %             T O P O F F
065534    %
065534    % SUBROUTINE TO CALL A ROUTINE IN PAGING-OFF AREA FROM THE PIT3 SEGMENT
065534    %
065534    % THIS ROUTINE MUST BE CALLED IN IOF OR WITH MONITOR LEVEL DISABLED
065534    % WHEN CALLED ON LEVEL 1.
065534    %
065534    SUBR TOPOFF
065534    INTEGER POINTER LREG,CROUTINE
065536    TOPOFF: X:=:L+1=:"LREG"-1; X:=X.S0=:"CROUTINE":=L; *POF
065546            CALL CROUTINE; GO NSKIP; MIN "LREG"
065551    NSKIP:  *PON
065552            GO LREG
065553    RBUS
065553
065553    %=============================================================================
065553    %             X T O P O F F
065553    %
065553    % PRIVATE SUBROUTINE FOR THE PFIXC ROUTINE ON PIT3 SEGMENT TO CALL
```

```
065553   % ROUTINES IN POF.
065553   %
065553   SUBR XTOPOFF
065553   INTEGER POINTER LREG,CROUTINE
065555   XTOPOFF: X:=:L+1=:"LREG"-1; X:=X.SO=:"CROUTINE":=L; *POF
065565            CALL CROUTINE; GO NSKIP; MIN "LREG"
065570   NSKIP:   *PON
065571            GO LREG
065572   RBUS
065572
065572   %=============================================================================
065572   %             S S 4 1 I P I T
065572   %
065572   % SUBROUTINE TO SET PIT3 SEGMENT INTO PIT3
065572   %
065572   % ENTRY: X=START OF PAGELINK
065572   %        T=CORMBANK
065572   %        B=ADDR OF FIRST PAGE TABLE ENTRY FOR PIT3 SEGMENT
065572   %
065572   SUBR SS41IPIT
065572   SS41IPIT: *PIOF
065573           DO
065573              *LDDTX DPGPR
065574              AD=:DSO; *LDXTX DPAGL
065576           WHILE X><0
065577              B+2
065600           OD; *PION
065602           EXIT
065603   RBUS
065603
065603   *"8SREE
"065603   INTEGER FIXCRT=?
065603   %=============================================================================
065603   % 36.15       S R E E N T
065603   %
065603   % MONITOR CALL SREENT(SEGNO) - ATTACH REENTRANT SEGMENT AND WRITE SHADDOW
065603   %                              PAGES WITH SWIP TO MASSTORAGE
065603
065603   SUBR SREENT
065603
065603   INTEGER SAVB,SGA,SGB,RSGA=?,RSTA=?,REND=?
065606   SREENT: CALL GET1; A:=B=:SAVB; *POF
065612           CALL XCSEGS              % CLEAR PAGE TABLES
065613           IF DO><0 THEN
065615              CALL LEGSEG; X=:RSGA; IF X=NSEGA OR X=NSEGB GO RETSTUPR
065625              IF A NBIT 5DEMAND GO ERRD
065627              CALL RLEGSEG
065630              T:=DO; NSEGA.FLAG; X=:SGA; CALL SHRSOVERLAP
065635              T:=DO; NSEGB.FLAG; X=:SGB; CALL SHRSOVERLAP
065642              CALL SRESER          % RESERVE SWAPPING AND DISC SYSTEM
065643              IF BACKGROUND><0 THEN CALL ESCOF FI     % DISABLE ESCAPE
065646              X:=RTREF=:FIXCRT
065650              CALL CLRBMAP; O=:RTREF.RSEGM; *POF
065654              RSGA.LOGADR/\377=:RSTA; X.LOGADR SHZ -10+RSTA=:REND
065664              IF X:=SGA><0 THEN CALL TFWBACK FI
065667              IF X:=SGB><0 THEN CALL TFWBACK FI
065672              SGA; T:=RSGA; CALL OVERLAP; SGB; T:=RSGA; CALL OVERLAP
065700              CALL SRELES; O=:FIXCRT; *PION          % RELEASE SWAPPING AND DISC SYSTEM
065703              IF BACKGROUND><0 THEN CALL ESCON FI
065706              SAVB=:B
```

```
065710          FI;
065710          X:=RTREF; CALL CLRBMAP; X:=RTREF
065713          X.RSEGM/\177400+D0=:X.RSEGM; GO RETSTUPR
065720
065720    % ERROR EXITS:
065720    ERRD:  CALL 9ERRA(#45); GO RETXIT  % NOT DEMAND SEGMENTS
065723
065723    *)FILL
065751
065751    %  LOCAL SUBROUTINE TO FIND SHADDOW PAGES WITH 5WIP AND WRITE THESE
065751    %  PAGES TO DISC.
065751    %  ENTRY:     X = SEGMENT TABLE ENTRY
065751    %             RSTA = FIRST LOGICAL PAGE FOR REENTRANT SEGMENT
065751    %             REND = LAST LOGICAL PAGE FOR REENTRANT SEGMENT
065751    %             SWAPPING AND DISCS MUST BE RESERVED!!
065751    %
065751    %  EXIT:      B IS DESTROID
065751
065751    INTEGER RSGA,RSTA,REND
065754    INTEGER POINTER LREG
065755    TFWBACK: A:=L=:"LREG"; X=:B; T:=0; X+1
065762          FLAG BZERO 50K=:FLAG
065765          DO; *LDXTX DPAGL
065766          WHILE X><0                          % FIND OVERLAPING PAGES
065767                T:=CORMBANK; *LDATX DALOG
065771                IF A>=RSTA AND A<REND THEN % SHADDOW PAGE
065777                   T:=CORMBANK; *LDATX DPGPR
066001                   IF A BIT 5WIP THEN A:=1=:D; B=:T; CALL ATRNSEG FI
066007                FI; T:=CORMBANK
066010          OD; GO LREG; *)FILL
066014
066014    RBUS
066014    *"
"066014
066014
066014    %=============================================================================
066014    %             S G A N D   S G O R
066014    %
066014    % SUBROUTINES TO THE SEGMENT-WRITE-PERMIT AND SEGMENT-WRITE-PROTECT COMMANDS
066014    % ENTRY: X=SEGMENT TABLE ENTRY; D=SEGMENT NUMBER
066014    % EXIT: ERROR
066014    % EXIT+1: OK
066014
066014    SUBR SGAND,SGOR
066014    INTEGER CSEG
066015    SGAND: K:="0"; GO FELLS
066017    SGOR:  K:=1
066020    FELLS: MLEV; *MCL PIE
066022          X=:CSEG
066023          IF X=SEGMA OR X=SEGMB OR X=SEGMC THEN
066034             D=:T; X.LOGADR/\377 SH 1+177000=:T
066042             X.LOGADR SHZ -10 SH 1; T=:X; T+A
066047             DO WHILE X><T
066051                *PIOF
066052                IF X.SO/\377=D THEN
066056                   X.SO; IF K THEN A BONE 5WPM ELSE A BZERO 5WPM FI; A=:X.SO
066065                FI; X+2; *PION
066067             OD
066070          FI; T:=0; X:=CSEG+1
066073          DO
```

```
066073              *LDXTX DPAGL
066074           WHILE X><0
066075              T:=CORMBANK
066076              *LDATX DPGPR
066077              IF K THEN A BONE 5WPM ELSE A BZERO 5WPM FI; *STATX DPGPR
066105           OD; CSEG.FLAG; IF K THEN A BONE 5WPM ELSE A BZERO 5WPM FI; A=:X.FLAG
066116           MLEV; *MST PIE
066120           EXITA
066121     RBUS
066127
066127
066127     %-============================================================================
066127     %              M X R E T
066127     %
066127     % ROUTINE TO RELEASE MONITOR CALL WORKING FIELD AND GO TO MRET
066127     %
066127     SUBR MXRET
066127     MXRET: MLEV; *MCL PIE
066131            X:=RTREF; CALL BRELEASE
066133            A:=B; *IRW MLEVB DX
066135            "MRET"; *IRW MLEVB DP
066137            "MONEN"; *IRW MLEVB DL
066141            MLEV; *MST PID; MST PIE; PION
066145            CALL ERRFATAL
066146     RBUS
066153
066153     *'8SIBA+8SIBX+8SIBM
"066153     %============================================================================
066153     % SPECIAL SIBAS MONITOR CALLS
066153     %
066153     *'8SIBA
"066153     *)KILL XSIB; XSIB=2
066153     *'8SIBX
"066153     INTEGER MXSIBAS(0); *XSIB
066154     *)KILL XSIB;
066154     *'8SIBA+8SIBX+8SIBM
"066154     INTEGER ARRAY  SIBBDEVS(0)
066154            *DSI0;DSI1;DSI2
066157     *'8SIBX+8SIBM
"066157     INTEGER ARRAY SIBAPDEVS(0)
066157            *DIA0;DIA1;DIA2
066162     *'8SIBX+8SIBM
"066162     *DIA0, 0;0;*-2;2;0;0;APLRS;*20/
066211     *DIA1, 0;0;*-2;2;0;0;APLRS;*20/
066240     *DIA2, 0;0;*-2;2;0;0;APLRS;*20/
066267     *DSI0, 0;0;*-2;2;*15/
066310     *DSI1, 0;0;*-2;2;*15/
066331     *DSI2, 0;0;*-2;2;*15/
066352
066352     *'8SIBX+8SIBM
"066352     'EDSIB=*
066352
066352     SUBR MAPSIB,MSIBB,3MAPSIB,3MSIBB
066352     MAPSIB: CALL GET0; CALL S3NOALPIT; GO 3MAPSIB
066355     MSIBB:  CALL GET0; CALL S3NOALPIT; GO 3MSIBB
066360     RBUS
066364
066364     *'
"066364
066364     *'7ENDC=*
```

```
066364
066364    %
066364    * -CXCPU
066364    @ELIB CXCPU
066364    @MAC

)9SCLC
066364    XX=3
066364          XZ=XX+3; )KILL XX; XX=XZ; )KILL XZ        % TERM 1
066364    "8TR5;  XZ=XX+3; )KILL XX; XX=XZ; )KILL XZ
066364    "8TR6;  XZ=XX+3; )KILL XX; XX=XZ; )KILL XZ
066364    "
066364    172000-XX/
171764    )KILL XX
171764    999EN=*
171764
171764    TIOBU=*
171764
171764                1;     56;     134
171767    "8TR5;     44;     56;     134
171772    "8TR6;     45;     56;     134
171775    "
171775    -1; -1; -1
172000    )9RCLC
)9SLPL172000    @ELIB
172000
172000    SUBR SETPTABL,INTDFIELDS
172000    RBUS
172000
172000    INTEGER TADFPHPAGE            % FIRST PHYSICAL PAGE IN TAD-STACK
172001    INTEGER TADLPHPAGE            % LAST PHYSICAL PAGE IN TAD-STACK
172002    INTEGER TDLPHPAGE             % LAST PHYSICAL PAGE IN TERMINAL-DATAFIELDS
172003
172003    %=============================================================================
172003    % 36.16       S I N T R
172003    %
172003
172003    INTEGER CCSTART=?,PIEREG=?,PIDREG=?,TMRTE=?,CCNOX=?
172003    INTEGER XCHREENTPAGES=?
172003    DOUBLE XCLEP=?,XSETP=?
172003    INTEGER XCLNREENT=?
172003    DOUBLE ARRAY NSWPAGE=?,NINIPAGE=?
172003    INTEGER IRTCPIT=?,77CBU=?
172003    DOUBLE ARRAY XCCTAB=?
172003    INTEGER FPS41=?
172003
172003    INTEGER FMEMCHECK:=0          % NO MEMORY TEST IN START-UP
172004
172004    SUBR SINTR
172004
172004    BASE SBFIELD
172004    INTEGER 9HDEV                 % SPECIAL IOX-NUMBER FOR MAIN SWAP-DEVICE
172005    INTEGER XIOBUTAB:=IOBUTAB     % ADDRESS OF I/O-DEVICES BUFFER TABLE
172006    INTEGER ARRAY LSWPDEV:=(502,1104,1100,1224,1231,1207)  % LOG.DEV. NO FOR SWAP-DEVICES
172014    INTEGER ARRAY HSWPDEV:=(500, 510,1540, 500, 510,1550)   % IOX-DEV. NO FOR SWAP-DEVICES
172022    INTEGER XSMRES:=9SMRE         % START OF PAGING-OFF CODE AREA
172023    INTEGER XEMRES:=9EMRE         % END OF PAGING-OFF AREA
172024    INTEGER CBUF:=9EMRE           % ADDRESS OF CURRENT I/O BUFFER
172025    INTEGER LPHYSPAGE             % LAST PAGE IN MEMORY
172026    INTEGER DBFNO:=5BUFA          % NUMBER OF DEVICE-BUFFER-HEADERS.
```

371 → 66131

400

532

```
172027    INTEGER MMSIZE                % SIZE OF MEMORY MAP
172030    INTEGER MMFPAGE:=-1           % FIRST PAGE OF MEMORY MAP
172031    INTEGER MMLPAGE:=-1
172032    INTEGER DBFPAGE:=-1           % FIRST PAGE USED AS DEVICE BUFFER
172033    INTEGER DBLPAGE:=-1           % LAST PAGE USED AS DEVICE BUFFER
172034    INTEGER FLAMPAGE:=0           % FIRST PHYSICAL PAGE FOR LAMU TABLES
172035    INTEGER LLAMPAGE:=0           % LAST PHYSICAL PAGE FOR LAMU TABLES
172036    INTEGER CLAMSIZE              % SIZE OF LAMU DESCRIPTION TABLE
172037    INTEGER NPLAMU:=1             % NUMBER OF PAGES USED BY LAMU TABLES
172040    INTEGER CURRPAGE              % CURRENT PAGE NUMBER TO CHECK
172041    INTEGER CHDLCF=CURRPAGE       % HDLC FLAG
172041    INTEGER CCOUNT
172042    DOUBLE ARRAY POINTER DCCTAB:=CCTAB
172043    INTEGER ARRAY POINTER ACCTAB=DCCTAB
172043    INTEGER ARRAY POINTER PNSWPAGE:=NSWPAGE
172044    INTEGER POINTER ICINDADDR(2)
172046    DOUBLE POINTER CINDADDR=ICINDADDR
172046    INTEGER POINTER FPGNO:=XFPGNO,PHYSPTEST:=XPHYSPTEST,GETAREA:=XGETAREA
172051    INTEGER POINTER INITPAGE:=XINIPAGE,MEMCHECK:=XMEMCHECK,OUTLINK:=XOUTLINK
172054    INTEGER XT,XA,XD,XXX; TRIPLE XTAD=XT
172060    INTEGER POINTER CLINK
172061    INTEGER PREVX,PREVT
172063    DOUBLE ARRAY POINTER PNINITPAGE:=NINITPAGE
172064    DOUBLE ARRAY POINTER PLAMARR:=LAMARR
172065    INTEGER 1CCTAB,2CCTAB
172067    INTEGER CINDX
172070    INTEGER NNRTP,POWFPAG,POWLPAG,POWSIZE
172074    DOUBLE DPOWADDR=POWFPAG
172074    INTEGER LPS41
172075    INTEGER BUSYMOD
172076    INTEGER SYMFPHPAGE,SYMLPHPAGE
172100
172100    @ICR
172100    INTEGER ARRAY ENTAB:=(EN000,0,EN200,EN300,EN400,EN500,EN600,EN700,
172110                          E1000,E1100,E1200,E1300,E1400,E1500,E1600,E1700,
172120                          E2000,E2100,E2200,E2300,E2400);
172125    @CR;
172125
172125    ESAB
172125
172125    DISP 0; INTEGER 5NBLCK,5RETRY; INTEGER POINTER 5TRNSF; PSID
172125    DISP -4; INTEGER M4; PSID
172125
172125
172125    % LOCAL SUBROUTINE TO FIND SPACE FOR THE I/O BUFFERS
172125    %
172125    % ENTRY:      A=0 : TRY TO PLACE ALL I/O BUFFERS (EXCEPT LARGE HDLC BUFFERS),
172125    %                 : IN POF.
172125    % EXIT:       I/O BUFFERS FOUND
172125    % EXIT+1:     NOT ENOUGH SPACE IN POF
172125    %
172125    % ENTRY:      A><0: SAME AS FOR A=0 EXCEPT THAT BUFFERS FOR SYNC.MODEM WILL
172125    %                   BE PLACED OUTSIDE POF
172125    % EXIT:       I/O BUFFERS OK
172125    %
172125    % EXIT+1:     FATAL ERROR
172125    %
172125    INTEGER CFIOFLG
172126    CFIOBUF: A=:CFIOFLG; 0=:BUSYMOD
172130           X:=XIOBUTAB; A:=CBUF=:D
```

```
172133          DO WHILE X.S0><-1
172137             IF A BIT BIHDLC AND X.S1/\77777>>=2000 THEN
172146                A:=0
172147             ELSE
172150                IF A BIT BISYMOD AND T:=CFIOFLG><0 THEN
172155                   X.S1/\77777+BUSYMOD=:BUSYMOD; GO NXT
172162                ELSE
172163                   X.S1/\77777
172165                FI
172165             FI; D+A; IF C THEN EXITA FI                % CROSS 64KW MEMORY BANK
172171  NXT:       X+2
172172          OD; IF D>>177000 THEN EXITA FI                % OVERLAP PAGE INDEX TABELS
172177          EXIT                                          % BUFFER IS FOUND
172200
172200  % LOCAL SUBROUTINE TO SIMULATE LOGPH FOR ELEMENTS IN IOBUTAB
172200  %
172200  % ENTRY:      A=LOGICAL DEVICE NUMBER
172200  %             CINDADDR MUST POINT TO ELEMENT IN IOBUTAB
172200  %
172200  % EXIT:       X=ADDRESS OF DATAFIELD
172200  %
172200  % CALL TO ERRFATAL WHEN ERROR
172200
172200  XLOGPH: AD SHZ -6; IF A=1 OR A>24 THEN CALL ERRFATAL FI
172210          A=:X:=D SHZ -12
172213          IF A+A+CNVRT(X)+1=:D-ENTAB(X)>=0 THEN CALL ERRFATAL FI
172222          AD:=D.DOUO; IF T:=ICINDADDR BIT 17 THEN D=:X ELSE A=:X FI
172232          IF X=0 THEN CALL ERRFATAL FI
172235          EXIT
172236  *)FILL
172243
172243  % INITIALIZE PAGING SYSTEM AND INTERRUPT SYSTEM
172243
172243  SINTR: A:=0; *PIOF; TRR IIE; TRA IIC
172247          A:=0; *TRR PID; TRR PIE
172252          A:=2; *TRR PCR; SEX; TRA STS
172256          IF A NBIT 5N100 THEN CALL ERRFATAL FI % THIS SYSTEM IS ONLY FOR NORD 100!!!!
172261          A=:CPSTA
172262          A:="LVO"; *IRW 0 DP; ION
172265  LVO:    *IOF
172266          "SBFIELD"=:B
172270  % INITIALIZE THE PAGE TABLES
172270          A:=162000; D:=0; X:=177000
172273          DO AD=:X.DOUO; D+1; X+2 WHILE X><0 OD
172300  % INITIALIZE THE MEMORY AREA FROM END OF RESIDENT UP TO 128 KBYTE
172300          X:=ENDCOR; DO X.S0=:X.S0 WHILE X><177777; X+1; OD
172310          A:=ENDCORE-1; CALL FPGNO
172313          IF A>"9POFS" SHZ -12 -1 THEN CALL ERRFATAL FI  % RESIDENT AND POF OVERLAPS
172321          A=:LRESP                                       % LAST PAGE IN RESIDENT
172322
172322  % TEST MEMORY ADDRESS BITS 19-23 (1-32MB)
172322  I>INTR: 1000=:CURRPAGE
172324  % IF MULTIPORT 3 THEN 3777=:ENDPAGE ELSE 37777=:ENDPAGE FI
172324          A:=200; *TRR IIE; TRA IIC; IOX 750; TRA IIC
172331          IF A=0 THEN A:=3777 ELSE A:=37777 FI; A=:ENDPAGE; A:=0; *TRR IIC
172340          DO WHILE CURRPAGE><ENDPAGE+1
172345             X:=0
172346             DO WHILE X<<"NINSZ*2"
172351                AD:=PNINITPAGE(X)
172352                IF A><0 AND A<<=CURRPAGE AND D>>=T GO NEXT
```

```
172360                  X+2
172361              OD; X:=0; 124000=:X.S0
172365              A:=CURRPAGE=:D:=162000; X:=177176; AD=:X.DOU0
172372              A:=1000; *TRR IIE; TRA IIC; PON
172376              X:=176000; X.S0; *POF; TRA IIC
172402              IF A=0 THEN
172403                  *PON
172404                  A:=X.S0=:D:=-1=:X.S0; *POF
172411                  X:=0
172412                  IF X.S0=124000 THEN
172416                      A:=D; X:=176000; *PON
172421                      A=:X.S0; *POF; TRA IIC
172424                  ELSE
172425                      124000=:X.S0; CURRPAGE-1=:ENDPAGE
172432                      GO LABL1
172433                  FI
172433              FI
172433  NEXT:       CURRPAGE SH 1=:CURRPAGE
172436          OD
172437
172437  % FIND LAST 32K MEMORY MODULE
172437  LABL1: A:=77=:LPHYSPAGE+1
172442  DO1:    DO WHILE A<<=ENDPAGE
172445              CALL PHYSPTEST; GO NOTEXIST; A+37=:LPHYSPAGE; A+1
172452          OD; GO L1
172454  NOTEXIST: A+40; GO DO1
172456  *)FILL
172477
172477  L1:     7EPOF=:CBUF
172501          A:=SEGSTART-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
172507          A=:NNRTP
172510  *"8XMSG
"172510            -1=:CBUF.S0; CBUF=:XZRTT+NNRTP=:CBUF
172517  *"
"172517           CBUF=:7RTDLGADDR
172521          A:=NNRTP*5XRTDSIZE+CBUF=:CBUF=:77CBU; IF C THEN CALL ERRFATAL FI
172631          GO LABL2; *)FILL
172642  % FIND THE I/O BUFFER AREA
172642  LABL2: A:=0; CALL FAR CFIOBUF; GO IOBOK
172645          A:=1; CALL FAR CFIOBUF; GO IOBOK; CALL ERRFATAL
172651  IOBOK:
172651  % CHECK IF "PAGING OFF" AREA AND I/O BUFFER AREA EXISTS IN MEMORY
172651          A:=0; CALL FPGNO; A=:LPOFP              % LAST PAGE IN "PAGING-OFF" AREA
172654          "9POFS"; CALL FPGNO; A=:FPOFP           % FIRST PAGE IN "PAGING-OFF" AREA
172657          DO
172657              CALL PHYSPTEST; CALL ERRFATAL; A+1
172662          WHILE A><LPOFP
172665          OD; GO LABL3; *)FILL
172674
172674  % SET UP THE I/O BUFFERS
172674
172674  LABL3: XIOBUTAB=:"CINDADDR"
172676          DO AD:=CINDADDR WHILE A><-1
172602              0=:CHDLCF
172603              IF A BIT BIHDLC THEN
172605                  A=:CHDLCF; IF D>>1777 GO NXIOD
172611              FI; A/\7777; CALL FAR XLOGPH
172613              CBUF=:X.BUFST; AD:=CINDADDR
172616              IF A:=D NBIT 17 THEN A SH 1 ELSE A BZERO 17 FI; A=:X.MAX=:X.CFREE
172626              IF CHDLCF><0 THEN 0=:MASTB FI
```

```
172631              A:=D BZERO 17+CBUF=:CBUF
172635    NXIOD:    "CINDADDR"+2=:"CINDADDR"
172640          OD
172641    % SET UP RT-COMMON TABLE
172641          IF CCNOX><0 THEN
172643             IF ACCTAB(0)=0 THEN
172646                IF CCSTART=0 THEN
172650                   LPHYSPAGE; X:=CCNOX; T:=LRESP+1
172654                   CALL VGETAREA; CALL ERRFATAL; A-CCNOX+1=:CCSTART
172661                FI; A=:D
172662                200-CCNOX+161000; T:=CCNOX; X:=0
172667                DO WHILE T><0
172671                   AD=:DCCTAB(X); A+1; D+1; X+2; T-1
172676                OD; -1=:ACCTAB(X)
172701             FI
172701          ELSE
172702             IF ACCTAB(0)=0 THEN -1=:ACCTAB(X) FI
172707          FI; GO L2; *)FILL
172722    % TEST IF RT-COMMON PAGES EXIST AND INITIALIZE THE RT-COMMON PAGES
172722    L2:   X:=0
172723          DO WHILE ACCTAB(X)><-1
172727             X+1; ACCTAB(X); CALL PHYSPTEST; CALL ERRFATAL
172733             CALL INITPAGE; X+1
172735          OD
172736          GO FILL1; *)FILL
172740    FILL1:
172740
172740    % FIND CONTINUOUS AREA FOR THE DEVICE-BUFFERS IN MEMORY BANK 1 TO 3,
172740    % AND INITIALIZE THE PAGES
172740    %
172740          A:=100; X:=DBFNO; T:=377; CALL GETAREA; CALL ERRFATAL      % LAST PAGE = 377
172745          A=:DBFPAGE/\77 SH 12=:BUFASTART            % START ADDRESS OF BUFFER AREA WITHIN BUFBANK
172751          A:=DBFPAGE SHZ -6=:BUFBANK                 % MEMORY BANK IN WHICH THE BUFFERS LIE
172754          DBFPAGE+DBFNO-1=:DBLPAGE
172760          FOR A:=DBFPAGE TO DBLPAGE DO CALL INITPAGE; CALL MEMCHECK; CALL ERRFATAL OD
172771
172771    % RESERVE MEMORY FOR THE LAMU SYSTEM AND CLEAR THE LAMU TABLES
172771
172771          IF GNLAMU><0 THEN
172773             A*LDTSZ=:CLAMSIZE
172775             A:=SEGSTART-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
173003             A*GNLPRT*ALMSZ+CLAMSIZE; CALL FPGNO; A+D=:NPLAMU
173011             A:=100; T:=LPHYSPAGE; X:=NPLAMU; CALL GETAREA; CALL ERRFATAL
173016             A=:D=:FLAMPAGE+NPLAMU-1=:LLAMPAGE:=0; AD SH 12=:DLAMDT
173026             A:=D+CLAMSIZE=:LAMACT
173031             A:=FLAMPAGE; T:=NPLAMU+A
173034             DO WHILE A><T; CALL ZEROPAGE; A+1; OD
173041             X:=0
173042             DO WHILE X<<"NINSZ*2"; AD:=PNINITPAGE(X)=:PLAMARR(X); X+2; OD
173051             GO AFLAM; *)FILL
173070          FI
173070
173070    AFLAM:
173070
173070    % RESERVE MEMORY FOR THE MONITOR-CALL-LOG TABLE
173070          IF "MCLGPAGE"><0 THEN
173072             A:=0; T:=LPHYSPAGE; X:=1; CALL GETAREA; CALL ERRFATAL
173077             A=:MCLGPAGE=:D:=0; AD SHZ 12; A=:MCLGBANK:=D=:TNMCALL
173106          FI
173106
```

```
173106    % RESERVE MEMORY FOR THE SYNC.MODEM BUFFERS IF THEY ARE PLACED
173106    % OUTSIDE MEMORY BANK #0
173106
173106          IF BUSYMOD><0 THEN
173110             CALL FPGNO; A+D=:X=:SYMLPHPAGE
173114             A:=0; T:=LPHYSPAGE; CALL GETAREA; CALL ERRFATAL
173120             A=:SYMFPHPAGE-1+SYMLPHPAGE=:SYMLPHPAGE
173124             A:=SYMFPHPAGE=:D:=0; AD SHZ 12; A=:MBSYMOD:=D=:CBUF
173133             XIOBUTAB=:"CINDADDR"
173135    % INITIALIZE THE SYNC.MODEM DATAFIELDS WITH
173135    % BUFFER START ADDR, MAX AND CFREE
173135             DO AD:=CINDADDR WHILE A><-1
173141                IF A BIT BISYMOD THEN
173143                   A/\7777; CALL FAR XLOGPH
173145                   CBUF=:X.BUFST
173147                   AD:=CINDADDR
173150                   IF A:=D NBIT 17 THEN A SH 1 ELSE A BZERO 17 FI
173156                   A=:X.MAX=:X.CFREE
173160                   A:=D BZERO 17+CBUF=:CBUF
173164                FI; "CINDADDR"+2=:"CINDADDR"
173167             OD
173170          ELSE
173171             0=:MBSYMOD
173172          FI; GO BYPFILL; *)FILL
173202    BYPFILL:
173202    @LIB CXCPU
173202    % COMPUTE REQUIRED SPACE FOR THE TERMINAL DATAFIELDS OUTSIDE RESIDENT
173202    % INITIALISE THIS PAGES
173202
173202          X:="TIOBU"+1; L:=0
173205          DO X.DSO WHILE A><-1
173211             A+D+TDISIZ+TDOSIZ; DO A-400; L+1 WHILE A>0 OD
173221             X+3
173222          OD
173223          A:=L*400; CALL FPGNO; A+D=:X=:TDLPHPAGE
173231          A:=0; T:=LPHYSPAGE; CALL GETAREA; CALL ERRFATAL
173235          A=:TDFPAGE SHZ -6=:TDFBANK; TDFPAGE-1+TDLPHPAGE=:TDLPHPAGE
173244    @ELIB
173244
173244    % RESERVE MEMORY FOR THE TAD-STACK
173244          T:=0; X:="D1400"+1
173247          DO
173247             IF "D1500"=X THEN X+1 FI                    % SEARCH IN LOG.DEV TABLES
173253          WHILE "9EBAD">>X
173256             AD:=X.DSO
173257             IF A><0 OR D><0 THEN T+1 FI
173263             X+2
173264          OD
173265          IF T><0 THEN                                   % RESERVE PAGES
173267             A:=T*TSSIZE; CALL FPGNO; A+D=:MMSIZE        % MMSIZE=NO.OF PAGES IN TAD-STACK
173274             X:=A; A:=0; T:=LPHYSPAGE; CALL GETAREA; CALL ERRFATAL
173301             A=:TADFPHPAGE-1+MMSIZE=:TADLPHPHAGE         % FIRST AND LAST PHYSICAL PAGE FOR TAD-STACK
173305             A:=TADFPHPAGE SHZ -6=:TSBANK                % MEMORY BANK FOR TAD-STACK
173310             A:=TADFPHPAGE SH 12=:CCOUNT                 % CCOUNT=ADDR OF STACK WITHIN A 64KW MEMORY BANK
173313             X:="D1400"+1=:L
173316             DO
173316                IF L="D1500" THEN L+1 FI                 % INITIALIZE THE TAD DATAFIELDS
173322             WHILE L<<"9EBAD"                            % WITH THE ADDR OF THE TAD-STACK AREA
173325                AD:=L.DSO
173327                IF A><0 OR D><0 THEN
```

```
173332                  T:=CCOUNT
173333                  IF A><0 THEN T=:A.TSTADD FI
173336                  IF D><0 THEN T=:D.TSTADD FI
173342                  CCOUNT+TSSIZE=:CCOUNT
173345               FI; L+1; L+1
173347            OD;
173350         FI; GO OVFILL; *)FILL
173366
173366   OVFILL:
173366   % COMPUTE SIZE OF THE PAGE-OWNER TABLE FOR MON SPLREE
173366         "NNBRT"*POTESIZE+1777 SHZ -12=:X=:POWSIZE; A:=0; T:=LPHYSPAGE
173376         CALL GETAREA; CALL ERRFATAL
173400         A=:D=:POWFPAGE+POWSIZE-1=:POWLPAGE:=0; AD SH 12=:DSREBADDR
173410
173410   % COMPUTE SIZE OF MEMORY MAP, FIND CONTINUOUS MEMORY FOR THE MEMORY MAP
173410   % AND SET UP MEMORY MAP FOR ALL PAGES (0 - LPHYSPAGE)
173410   % THE PHYSICAL PAGE NUMBER IS INDEX IN THE MEMORY MAP
173410         IF LPHYSPAGE=37777 THEN A-1=:LPHYSPAGE FI
173416         A+2 SH 2; CALL FPGNO; A+D=:MMSIZE
173423         100; X:=MMSIZE; T:=LPHYSPAGE; CALL GETAREA; CALL ERRFATAL
173430         A=:D=:MMFPAGE SHZ -6=:CORMBANK:=D+MMSIZE-1=:MMLPAGE
173440         IF A:=D/\77=0 THEN A SH 12+4 ELSE A SH 12 FI
173447         A=:CORMSTART=:"XSGRT".BPAGLINK=:D:=CORMBANK; AD=:DCORMSTART
173455         LPHYSPAGE=:ENDPAGE
173457         GO L3; *)FILL
173473   L3:   D=:CURRPAGE; X:=CORMSTART
173475         FOR CURRPAGE DO WHILE CURRPAGE<<=LPHYSPAGE
173501            T:=CORMBANK; A:=X+4; D:=0; *STDTX DPAGL
173506            CURRPAGE=:D; A:=1; *STDTX DPGPR
173512            X+4
173513         OD; X-4=:ECORMAP; T:=CORMBANK; *STZTX
173521
173521   % RESERVE MEORY FOR PIT3 SEGMENT
173521         "XLPT3"+1-"XSPT3" SHZ -12=:MMSIZE
173526         "SG41".LOGADR/\377; T:=MMSIZE SH 10; A\/T=:X.LOGADR
173535         X:=MMSIZE; A:=100; T:=LPHYSPAGE; CALL GETAREA; CALL ERRFATAL
173542         A=:D=:FPS41+MMSIZE-1=:LPS41
173547         IF "SG33".LOGADR/\77 SH 12><0 AND A<<="XLPT3" THEN
173557            CALL ERRFATAL                          % PIT3 SEG AND X-MESSAGE OVERLAPS
173560         FI
173560   @LIB CXCPU-,
173560   @LIB CXCPU
173560         A:=LRESP=:D:=0; CALL OUTLINK               % LINK OUT RESIDENT
173564   @ELIB
173564         A:=LPOFP=:D:=FPOFP; CALL IOUTLINK          % LINK OUT POF AREA
173570         A:=MMLPAGE=:D:=MMFPAGE; CALL IOUTLINK      % LINK OUT MEMORY MAP AREA
173574         A:=DBLPAGE=:D:=DBFPAGE; CALL IOUTLINK      % LINK OUT DEVICE BUFFER AREA
173600         A:=POWLPAGE=:D:=POWFPAGE; CALL IOUTLINK    % LINK OUT PAGE-OWNER TABLE
173604         IF "MCLGBANK"><0 THEN
173606            A:=MCLGPAGE=:D; CALL IOUTLINK           % LINK OUT MONITOR-CALL-LOG-TABLE PAGE
173611         FI
173611         IF TADLPHPAGE><0 THEN
173613            A=:D:=TADFPHPAGE; CALL IOUTLINK         % LINK OUT TAD-STACK PAGES
173616         FI
173616         IF SYMLPHPAGE><0 THEN
173620            A=:D:=SYMFPHPAGE; CALL IOUTLINK         % LINK OUT SYNC.MODEM I/O BUFFERS
173623         FI
173623   @LIB CXCPU
173623         A:=TDLPHPAGE=:D:=TDFPAGE; CALL IOUTLINK    % LINK OUT TERMINAL DATAFIELD AREA
173627   @ELIB
```

```
173627          A:=LPS41=:D:=FPS41; CALL OUTLINK            % LINK OUT PAGES RESERVED FOR PIT3 SEGMENT
173633          IF GNLAMU><0 THEN
173635             A:=FLAMPAGE+NPLAMU-1=:D:=FLAMPAGE
173642             CALL OUTLINK                             % LINK OUT LAMU TABLE PAGES
173643          FI; A:=DBFPAGE SH 2+CORMSTART=:BUFMAPSTA
173647          GO L31; *)FILL
173677
173677    % LINK OUT RT-COMMON
173677    L31:  IF ACCTAB(0)><-1 THEN
173704             A/\377=:ARTFPAGE SH 1; T:="XCCTAB"-A=:IRTCPIT
173712             177=:ARTLPAGE
173714             DO WHILE ACCTAB(X)><-1
173720                X=:CINDX; A=:1CCTAB
173722                X+1; ACCTAB(X)=:D=:2CCTAB; CALL IOUTLINK
173727                A:=2CCTAB=:D:=161000; AD=:XCCTAB(CINDX); X+2
173735             OD
173736          ELSE
173737             -1=:ARTFPAGE=:ARTLPAGE
173742          FI; IF LPOFP><77 THEN A:=77=:D; CALL OUTLINK FI      % LINK OUT PAGE #77
173751    % LINK OUT NOT-SWAP PAGES AND INITIALIZE THE PAGES
173751          X:=0
173752          DO WHILE X<<"NNSWSZ*2"
173755             AD:=NSWPAGE(X)
173756             IF A><0 AND A<<=D THEN
173761                CALL OUTLINK
173762                DO WHILE A<<=D; CALL INITPAGE; A+1; OD
173767             FI; X+2
173770          OD
173771    % FIND NUMBER OF SWAPPING PAGES AND INITIALIZE THE SWAPPING PAGES
173771          0=:CURRPAGES; X:="XSGRT"+1=:PREVX; T:=0=:PREVT
173777    INDO2: DO
173777             *LDXTX DPAGL
174000          WHILE X><0
174001             T:=CORMBANK; *LDATX DPAGP
174003             CALL PHYSPTEST; GO COUTLINK; CALL INITPAGE
174006             CALL MEMCHECK; GO ERRPAGE; MIN CURRPAGE
174011             T=:PREVT; X=:PREVX
174013          OD; CURRPAGE=:NOPGS; GO L4
174017    ERRPAGE: CALL 9ERR(#03)
174021    COUTLINK: T:=CORMBANK; *LDATX DPAGL
174023          T:=PREVT; X=:D:=PREVX; *STATX DPAGL
174027          X:=D; T:=CORMBANK; A:=0=:D; *STDTX DPAGL; STZTX DPGPR
174035          X:=PREVX; T:=PREVT; GO INDO2
174040    *)FILL
174055
174055    L4:   "MASKE"=:PIEREG; 0=:PIDREG
174060          CALL XCHIOX; XTMRTERM=:TMRTERM
174063          CALL SYSEVAL
174064    @LIB CXCPU-,
174064    @LIB CXCPU
174064          IF HWINFO(0)/\377><2 THEN CALL ERRFATAL FI           % MUST BE ND-100/CX
174073    @ELIB
174073          ESGTABLE-SEGSTART=:D:=0; T:=5SEGSIZE; *RDIV ST
174101          A-1=:SGMAX; "SEGTX"-SEGSTART=:D:=0; *RDIV ST
174110          A=:USEGM
174111    % SET TIMESLICE CLASS
174111    % IF SAVED TIMESLICE CLASS (TSLSTATUS BIT 15-17) EQUALS ZERO THEN
174111    %           TIMESLICE CLASS FOR TERMINALS IS SET TO ZERO AND TIMESLICE
174111    %           CLASS FOR BATCH IS SET TO ONE
174111    % ELSE
```

```
174111    %           TIMESLICE CLASS IS SET EQUAL SAVED TIMESLICE CLASS
174111    % ENDIF
174111    %
174111            FPTSLICE=:B; D:=0
174114            DO WHILE B<<LPTSLICE
174117               IF TSLSTATUS(D)/\160000=0 THEN
174123                  X:="BCHTA"+1
174125                  DO WHILE X.S0><-1
174131                     IF X.S1=B THEN T:=22000; GO BBBAT FI        % BATCH
174136                     X+4
174137                  OD; T:=0
174141               ELSE
174142                  T:=A SHZ -3\/A
174145    BBBAT:     FI; A:=TSLSTATUS(D)/\1000\/T=:TSLSTATUS(X)
174152               A:=SRTSIZE; B+A; D+1
174155            OD;
174156    * 8BFD1 8FDI1
174156     GO INTDFIELDS
174157
174157    RBUS
174206
174206    %%@LIB CXCPU
174206    %=============================================================================
174206    %       X C H I O X
174206    %
174206    % CHECK FOR IOX ERRORS ON THE TERMINALS
174206    %
174206    SUBR XCHIOX
174206
174206    INTEGER POINTER LREG,PTMRTE
174210    INTEGER XREG; TRIPLE TADREG
174214    INTEGER CCOUNT
174215
174215    XCHIOX: TAD=:TADREG; X=:XREG; A:=L=:"LREG"
174221            A:=200; *TRR IIE; TRA IIC
174224            "TMRTE"+1=:"PTMRTE"
174227            DO WHILE X:=PTMRTE><-1 AND "PTMRTE"<<"9EXTD"
174237               *TRA IIC
174240               IF X><0 AND X.HDEV><0 THEN
174243                  T:=A+2; *IOXT; TRA IIC
174247                  IF A=7 THEN
174252                     0=:PTMRTE
174253                  ELSE
174254                     IF X.HDEV NBIT 2 THEN
174257    % CHECK IDENT CODE IF INPUT DEVICE
174257                        X=:D; T:=X.HDEV+3; A:=30; *IOXT
174264                        *IDENT PL10; IDENT PL12; TRA IIC
174267                        15; *IOXT
174271                        A:=0; T+"5-3"; *IOXT
174274                        0=:CCOUNT; FOR CCOUNT DO OD; FOR CCOUNT DO OD
174301                        5; T+"7-5"; *IOXT
174304                        A:=0; *IDENT PL12
174306                        IF A=0 GO ERRID; A-1
174310                        IF A<<="HIDTERM" AND A>>="LIDTERM" THEN
174316                           IF A-T>>"MXX12" GO ERRID
174322                           X:="ID12T"+A; X.S0
174325                        ELSE
174326                           ITB12(A)
174330                        FI; IF A><D GO ERRID; A:=0; *IDENT PL10
174334                        IF A=0 GO ERRID: A-1
```

```
174336                      IF A<<="HIDTERM" AND A>>="LIDTERM" THEN
174344                         IF A-T>>"MXX10" GO ERRID
174350                         X:="ID10T"+A; X.SO
174353                      ELSE
174354                         ITB10(A)
174356                      FI
174356                      T:="9CXTI"; D+T
174360                      IF A><D GO ERRID
174362                      T+"3-7"; A:=30; *IOXT; IDENT PL10; IDENT PL12; TRA IIC
174370                      GO IDOK
174371   ERRID:             D.HDEV; T:=0; CALL 9ERR(#23)
174376   IDOK:           FI
174376                FI
174376             FI; MIN "PTMRTE"
174377          OD; TAD:=TADREG; X:=XREG; GO LREG
174403   RBUS
174420
174420
174420   %=============================================================================
174420   % 36.18       X G E T A R E A     Y G E T A R E A
174420   %
174420   % SUBROUTINE TO ALLOCATE CONTINUOUS AREA IN MEMORY
174420   %
174420   % ENTRY: A=FIRST PAGE TO TEST
174420   %        B=SBFIELD IN SINTR
174420   %        X=NUMBER OF PAGES IN AREA
174420   %        T=LAST POSSIBLE PAGE IN AREA
174420   %
174420   %        XGETAREA: ALLOCATES PAGES FROM A UPP TO T
174420   %                  AND TEST THAT WHOLE AREA IS IN ONE BANK
174420   %        YGETAREA: ALLOCATES PAGES FROM A DOWN TO T
174420   %                  (CALLED WHEN ALLOCATING PAGES TO RT-COMMON)
174420   % EXIT:  AREA NOT FOUND
174420   %
174420   % EXIT+1: AREA FOUND, A=FIRST PAGE IN AREA
174420   %
174420
174420   % N O T E: THIS DISP DEFINITION MUST CORRESPOND TO THE BASEFIELD IN SINTR
174420
174420   SUBR XGETAREA,YGETAREA
174420
174420   DISP -200
174420   INTEGER 9HDEV                  % SPECIAL IOX-NUMBER FOR MAIN SWAP-DEVICE
174420   INTEGER XIOBUTAB               % ADDRESS OF I/O-DEVICES BUFFER TABLE
174420   INTEGER ARRAY LSWPDEV(6)       % LOG.DEV. NO FOR SWAP-DEVICES
174420   INTEGER ARRAY HSWPDEV(6)       % IOX-DEV. NO FOR SWAP-DEVICES
174420   INTEGER XSMRES                 % START OF PAGING-OFF CODE AREA
174420   INTEGER XEMRES                 % END OF PAGING-OFF AREA
174420   INTEGER CBUF                   % ADDRESS OF CURRENT I/O BUFFER
174420   INTEGER LPHYSPAGE              % LAST PAGE IN MEMORY
174420   INTEGER DBFNO                  % NUMBER OF DEVICE-BUFFER-HEADERS.
174420   INTEGER MMSIZE                 % SIZE OF MEMORY MAP
174420   INTEGER MMFPAGE                % FIRST PAGE OF MEMORY MAP
174420   INTEGER MMLPAGE
174420   INTEGER DBFPAGE                % FIRST PAGE USED AS DEVICE BUFFER
174420   INTEGER DBLPAGE                % LAST PAGE USED AS DEVICE BUFFER
174420   INTEGER FLAMPAGE               % FIRST PHYSICAL PAGE FOR LAMU TABLES
174420   INTEGER LLAMPAGE               % LAST PHYSICAL PAGE FOR LAMU TABLES
174420   INTEGER CLAMSIZE               % SIZE OF LAMU DESCRIPTION TABLE
174420   INTEGER NPLAMU                 % NUMBER OF PAGES USED BY LAMU TABLES
```

```
174420    INTEGER CURRPAGE              % CURRENT PAGE NUMBER TO CHECK
174420    INTEGER CCOUNT
174420    DOUBLE ARRAY POINTER DCCTAB
174420    INTEGER ARRAY POINTER ACCTAB=DCCTAB
174420    INTEGER ARRAY POINTER PNSWPAGE
174420    INTEGER POINTER ICINDADDR(2)
174420    DOUBLE POINTER CINDADDR=ICINDADDR
174420    INTEGER POINTER FPGNO,PHYSPTEST,GETAREA
174420    INTEGER POINTER INITPAGE,MEMCHECK,XYOUTLINK
174420    INTEGER XT,XA,XD,XXX; TRIPLE XTAD=XT
174420    INTEGER POINTER CLINK
174420    INTEGER PREVX,PREVT
174420    DOUBLE ARRAY POINTER PNINITPAGE
174420    DOUBLE ARRAY POINTER PLAMARR
174420    INTEGER 1CCTAB,2CCTAB
174420    INTEGER CINDX
174420    INTEGER NNRTP,POWFPAG,POWLPAG,POWSIZE
174420    INTEGER LPS41
174420    INTEGER BUSYMOD
174420    INTEGER SYMFPHPAGE,SYMLPHPAGE
174420    PSID
174420
174420    DOUBLE ARRAY POINTER PCUMTABLE:=CUMTABLE
174421    YGETAREA: K:="0"; GO L1
174423    XGETAREA: K:=1
174424    L1:     TAD=:XTAD; X-=:XXX; T:=L=:"CLINK"
174431    AGAIN:  XXX=:CCOUNT; XA=:CURRPAGE
174435            FOR CCOUNT DO
174435               IF CURRPAGE<<100 THEN
174441                  IF A<<=LRESP GO FAR TRNX
174444                  IF A=77 GO FAR TRNX                    % DO NOT USE PAGE 77
174447                  IF A>>=FPOFP AND A<<=LPOFP GO FAR TRNX  % FOR DEVICE BUFFERS OR CORE MAP
174455               FI; IF A>>=DBFPAGE AND A<<=DBLPAGE GO FAR TRNX
174463               IF A>>=TDFPAGE AND A<<=TDLPHPAGE GO FAR TRNX
174471               IF A>>=POWFPAGE AND A<<=POWLPAGE GO FAR TRNX
174477               IF A>>=FLAMPAGE AND A<<=LLAMPAGE GO FAR TRNX      % IN LAMU TABLE AREA?
174505               IF A>>=MMFPAGE AND A<<=MMLPAGE GO FAR TRNX        % MEMORY MAP AREA
174513               IF T:="MCLGPAGE"><0 AND A=MCLGPAGE GO FAR TRNX    % MONCALL LOG PAGE
174521               IF A>>=TADFPHPAGE AND A<<=TADLPHPAGE GO FAR TRNX % TAD STACK AREA
174527               IF A>>=FPS41 AND A<<=LPS41 GO FAR TRNX            % MEMORY RESERVED FOR PIT3 SEGMENT
174535               IF A>>=SYMFPHPAGE AND A<<=SYMLPHPAGE GO FAR TRNX % SYNC.MODEM I/O BUFFER
174543               X:=0
174544               IF K THEN
174546                 DO WHILE ACCTAB(X)><-1                        % RT COMMON
174552                    X+1; IF ACCTAB(X)=CURRPAGE GO TRNX; X+1
174560                 OD; X:=0
174562               FI
174562               DO WHILE X<<"NINSZ*2"
174565                  AD:=PNINITPAGE(X)
174566                  IF A><0 AND A<<=CURRPAGE AND D>>=T GO TRNX
174574                  X+2
174575               OD; X:=0
174577               DO WHILE X<<"CUMSIZE*2"
174602                  AD:=PCUMTABLE(X)
174603                  IF A><0 AND A<<=CURRPAGE AND D>>=T GO TRNX
174611                  X+2
174612               OD
174613               CURRPAGE; CALL PHYSPTEST; GO TRNX
174616               IF K THEN
174620                  CURRPAGE/\177700=:T; IF XA/\177700><T GO TRNX     % CHECK THAT WHOLE
```

```
174627                MIN CURRPAGE                                    % AREA IS IN ONE BANK
174630              ELSE
174631                CURRPAGE-1=:CURRPAGE
174634              FI
174634          OD; GO OUT; *)FILL
174652  OUT:    MIN "CLINK"; A:=XA; GO CLINK          % AREA FOUND
174655  TRNX:   A:=XXX;
174656          IF K THEN A-; IF A+CURRPAGE >> XT GO CLINK; CURRPAGE+1=:XA;
174670          ELSE          IF A+CURRPAGE << XT GO CLINK; CURRPAGE-1=:XA;
174700          FI;GO FAR AGAIN
174701
174701  RBUS
174702
174702
174702  %=============================================================================
174702  %
174702  % SUBROUTINE TO INITIALIZE A PAGE (READ AND WRITE IN EVERY ADDRESS IN THE PAGE)
174702  %
174702  % ENTRY: A=PAGE NUMBER
174702  %
174702  % 36.19       X I N P A G E
174702  %
174702  SUBR XINIPAGE
174702  INTEGER XREG,TREG,AREG,DREG
174706  TRIPLE TADR=TREG
174706  XINIPAGE: TAD=:TADR; X=:XREG
174710          X:=0
174711          DO WHILE X<<"NINSZ*2"
174714             AD:=NINITPAGE(X)
174715             IF A><0 AND A<<=AREG AND D>>=T GO OUT
174723             X+2
174724          OD
174725          A:=AREG=:D:=162000; X:=177176; AD=:X.DOU0; *PON
174733          FOR X:=176000 DO X.SO=:X.SO OD; *POF
174740  OUT:    TAD:=TADR; X:=XREG; EXIT
174743  RBUS
174747
174747  %=============================================================================
174747  % 36.20       X M E M C H E C K
174747  %
174747  %
174747  % SUBROUTINE TO DO MEMORY CHECK ON A PAGE
174747  %
174747  % ENTRY: A=PAGE NUMBER
174747  %
174747  % EXIT:  MEMORY ERROR IN PAGE
174747  % EXIT+1: OK
174747  %
174747  SUBR XMEMCHECK
174747  *"8MEMC
"174747   XMEMCHECK:
174747  *"-8MEMC
"174747           EXITA
174750  *"8MEMC
"174750   RBUS
174750
174750  %=============================================================================
174750  % 36.21       X F P G N O
174750  %
174750  %
```

```
174750    % SUBROUTINE TO CONVERT ADDRESS TO PAGE NUMBER
174750    %
174750    % ENTRY: A=ADDRESS
174750    %
174750    % EXIT:   A=PAGE NUMBER
174750    %         D=0 WHEN ADDRESS IS MODULO 2K BYTE ELSE D=1
174750    %
174750    SUBR XFPGNO
174750    INTEGER SAVD
174751    XFPGNO: D:=0; AD SHZ -12; IF D><0 THEN D:=1 FI; EXIT
174757    RBUS
174757
174757
174757    %=============================================================================
174757    % 36.11       X P H Y S P T E S T
174757    %
174757    % SUBROUTINE TO TEST IF PAGE EXIST IN MEMORY
174757    %
174757    % ENTRY: A=PAGE NUMBER
174757    %
174757    % EXIT:   PAGE DOES NOT EXIST
174757    %
174757    % EXIT+1: PAGE EXSIST
174757    %
174757    SUBR XPHYSPTEST
174757    INTEGER XREG,TREG,AREG,DREG
174763    TRIPLE TADR=TREG
174763    XPHYSPTEST: TAD=:TADR; X=:XREG:=0
174766            DO WHILE X<<"NINSZ*2"
174771               AD:=NINITPAGE(X)
174772               IF A><0 AND A<<=AREG AND D>>=T GO NOTOK
175000               X+2
175001            OD; A:=AREG=:D:=162000; X:=177176; AD=:X.DOUO; A:=1000
175010            *TRR IIE; PON; TRA IIC
175013            X.SO; *POF; TRA IIC
175016            IF A=0 THEN L+1 FI; A:=0; *TRR IIE
175022    NOTOK: TAD:=TADR; X:=XREG; EXIT
175025    RBUS
175031
175031    %=============================================================================
175031    %             X O U T L I N K   -   I O U T L I N K
175031    %
175031    % SUBROUTINE TO LINK OUT AN AREA FROM THE MEMORY MAP
175031    % IOUTLINK WILL ALSO INITIALIZE THE AREA (READ AND WRITE IN ALL LOCATIONS)
175031    %
175031    % ENTRY:      A=FIRST PAGE TO LINK OUT
175031    %             D=LAST PAGE TO LINK OUT
175031    SUBR XOUTLINK,IOUTLINK
175031    INTEGER TREG,AREG,DREG,XREG,PREVX,PREVT
175037    TRIPLE TADR=TREG
175037    INTEGER POINTER LREG
175040    IOUTLINK: K:=1; GO FELLS
175042    XOUTLINK: K:="0"
175043    FELLS: TAD=:TADR; X=:XREG:=L=:"LREG"
175047            IF K THEN
175051               DO CALL XINIPAGE; WHILE A><D; A+1; OD; TAD:=TADR
175057            FI; X:="XSGRT"+1; T:=0
175062            DO
175062               IF X=0 THEN CALL ERRFATAL FI
175065               X=:PREVX; T=:PREVT
```

```
175067              *LDXTX DPAGL
175070              T:=CORMBANK; *LDATX DPAGP
175072              IF A-AREG=0 GO FOUND
175074          OD
175075    FOUND: DO
175075              *LDATX DPAGP
175076              IF A-DREG=0 THEN
175100                  *LDATX DPAGL
175101                  X=:D:=PREVX; T:=PREVT; *STATX DPAGL
175105                  X:=D; T:=CORMBANK; A:=0=:D; *STDTX DPAGL; STZTX DPGPR
175113                  TAD:=TADR; X:=XREG; GO LREG
175116              FI; A:=0=:D; *STDTX DPAGL; STZTX DPGPR
175122              X+4
175123          OD
175124    RBUS
175130    %=============================================================================
175130    %       Z E R O P A G E
175130    %
175130    % SUBROUTINE TO FILL A PAGE WITH ZEROES
175130    %
175130    % ENTRY:       A=PHYSICAL PAGE NUMBER
175130    %
175130    SUBR ZEROPAGE
175130    INTEGER XREG; TRIPLE TADR
175134    ZEROPAGE: TAD=:TADR; X=:XREG
175136          A=:D;=162000; X:=177176; AD=:X.DOU0; *PON
175143          FOR X:=176000 DO 0=:X.S0 OD; *POF
175147          TAD:=TADR; X:=XREG; EXIT
175152    RBUS
175155
175155    @LIB CXCPU
175155    INTEGER PSTWRI:=0              % ADDRESS OF BUFF.TERM DRIVER IF INCLUDED
175156
175156    %=============================================================================
175156    %               M A K E T D F S
175156    %
175156    % SUBROUTINE TO THE INTDFIELDS ROUTINE
175156    %
175156    SUBR MAKETDFS
175156
175156    INTEGER 1CDFADDR=?,2CDFADDR=?; DOUBLE DCDFADDR=?
175156    INTEGER TRLND=?,BINSZ=?,BOUSZ=?; TRIPLE TIOBE=?; DOUBLE TBUSZ=?
175156    DISP 0; TRIPLE DTIOBE; PSID
175156    INTEGER LREG,CINDX
175160
175160    MAKETDFS: A:=:L+1=:LREG; X=:CINDX
175164          A-1; TAD:=A.S0.DTIOBE=:TIOBE
175171          A:=TDFPAGE=:D:=0; AD SHZ 12; D+L; A:=A+C; AD=:DCDFADDR
175200          FOR X:="XTDFELT" TO "XEDFELT" DO 0=:X.S0 OD
175207          CINDX=:B; IF TYPRING NBIT 5TERM THEN CALL ERRFATAL FI
175215          X:="XTDFELT"+TDINDI
175217          -1=:X.RSISTE; ZROUSPEC=:X."ROUSPEC"; ZCTTYP=:X.CTTYP
175225          ZCESCP=:X.CESCP; ZTSPEED=:X.TSPEED; ZCNTREG=:X.CNTREG
175233          ZDFLAG=:X.DFLAG; ZDBPROG=:X.DBPROG; "ECHO"=:X.ECHOTAB
175241          "BRKO"=:X.BRKTAB; "TTIMR"=:X."TMSUB"; -5=:X.TMR=:X.TTMR
175250          HDEV=:X.HDEV; TYPRING=:X.TYPRING; A:=B=:X.TDRADDR
175256          "TYENT"=:X."DRIVER"; "STTIN"=:X."STDRIV"
175262          "TTGET"=:X."IOTRANS"; "TEXIT"=:X."STDEV"; "CTRTI"=:X."SETDV"
175270          A:=2CDFADDR/\1777+"SUBFPAGE*2000"+TDISIZ+BINSZ+TDONDI=:X.DFOPP
175277          A:=2CDFADDR/\1777+TDISIZ=:X.BUFST
```

```
175303          BINSZ+BINSZ=:X.MAX=:X.CFREE
175307          CINDX+"9CXTI"=:X.XDFOPP
175312          X:=1CDFADDR; T:=2CDFADDR; A:=TDISIZ=:L:="XTDFELT"=:D:=0; *MOVPP
175322          A:=2CDFADDR+TDINDI=:TDFLGADDR
175325          AD:=DCDFADDR SHZ -12; A:=D=:TDFPHPAGE
175331          0=:RESLINK=:RTRES; A:=B=:BWLINK; 0=:ISTATE=:MLINK
175337          "STTIN"=:"STDRIV"; "TYENT"=:"DRIVER"
175343          GO L1; *)FILL
175363
175363  INTEGER 1CDFADDR,2CDFADDR; DOUBLE DCDFADDR=1CDFADDR
175365  INTEGER TRLND,BINSZ,BOUSZ; TRIPLE TIOBE=TRLND; DOUBLE TBUSZ=BINSZ
175370  INTEGER POINTER PLREG:=LREG, PCINDX:=CINDX
175372  INTEGER CDRIV
175373
175373  L1:     FOR X:="XTDFELT" TO "XEDFELT" DO 0=:X.S0 OD
175402          PCINDX+"9CXTI"=:B
175405          IF TYPRING NBIT 5TERM THEN CALL ERRFATAL FI
175411          X:="XTDFELT"+TDONDI
175413          1=:X.EMPTFLAG; "TTOMR"=:X."TMSUB"; -10=:X.TTMR
175421          HDEV=:X.HDEV; "TRTPUT"=:X."IOTRANS"; "DMOUT"=:X."STDEV"
175427          TYPRING=:X.TYPRING; T:=HDEV+DST; *IOXT     % READ STATUS
175434          IF A BIT 10 AND PSTWRI><0 THEN; ELSE "DWRITE" FI
175442          A=:CDRIV=:X."DRIVER"=:X."STDRIV"
175445          "CTRTO"=:X."SETDV"; A:=2CDFADDR/\1777+"SUBFPAGE*2000"+TDINDI=:X.DFOPP
175454          A:=2CDFADDR/\1777+TDISIZ+TDOSIZ+BINSZ=:X.BUFST
175462          BOUSZ+BOUSZ=:X.MAX=:X.CFREE SHZ -1=:X.MINBHOLD
175470          A:=B=:X.TDRADDR; PCINDX=:X.XDFOPP
175474          X:=1CDFADDR; A:=2CDFADDR+TDISIZ+BINSZ=:T; A:=TDOSIZ=:L:="XTDFELT"=:D:=0; *MOVPP
175507          A:=2CDFADDR+TDISIZ+TDONDI+BINSZ=:TDFLGADDR
175514          AD:=DCDFADDR SHZ -12; A:=D=:TDFPHPAGE
175520          0=:ISTATE=:RESLINK=:RTRES=:MLINK; A:=B=:BWLINK
175526          CDRIV=:"DRIVER"=:"STDRIV"
175531          AD:=DCDFADDR; T:=400; D+T; A:=A+C; AD=:DCDFADDR
175536          PLREG=:P
175540  *)FILL
175555  INTEGER ARRAY XTDFELT(120),XEDFELT(1)
175676  RBUS
175676
175676
175676
175676  @ELIB
175676
175676  * 99END=*
175676
175676  @DEV 1
175676  @DEV (S-S-J)MRES-SINI
```

```
175676
175676   %=============================================================================
175676   %      M R E S - S I N I
175676   %=============================================================================
175676
175676
175676   *9SMRE=9POFS
175676   *9SMRE/
106000
106000   @LIB CXCPU
106000   %=============================================================================
106000   % 11.19          B R K 0   B R K 1   B R K 2       B R K 3   B R K 4   B R K 5
106000   %               E C H 0   E C H 1   E C H 2       E C H 3   E C H 4   E C H 5
106000   %               B R K T B   E C H T B
106000   %
106000
106000   %BREAK TABLES:
106000   INTEGER ARRAY BRK0:=(-1,-1,-1,-1,-1,-1,-1,-1)                  % BREAK ON ALL
106010   INTEGER ARRAY BRK1:=(177737,-1,0,0,0,0,0,1)                    % CONTROL
106020   INTEGER ARRAY BRK2:=(-1,-1,45003,41,100000,11,-1,-1)           % MAC
106030   INTEGER ARRAY BRK3:=(-1,-1,-1,100077,-1,-1,-1,-1)              % NO BREAK ON 1 - 9
106040   INTEGER ARRAY BRK4:=(-1,-1,77777,-1,100000,37,100000,37)       % NO BREAK ON LETTER
106050   INTEGER ARRAY BRK5:=(-1,-1,0,0,0,0,0,3)                        % NO BREAK ON ALPHANUMERIC
106060   INTEGER ARRAY BRK6:=(4,0,0,0,0,0,0,0)                          % BREAK ONLY ON ( CR. )
106070
106070   %ECHO TABLES:
106070   INTEGER ARRAY ECH0:=(0,0,0,0,0,0,0,1)
106100   INTEGER ARRAY ECH1:=(177377,-1,0,0,0,0,0,1)
106110   INTEGER ARRAY ECH2:=(177333,-1,0,0,0,0,0,1)
106120   INTEGER ARRAY ECH3:=(-1,-1,-1,100077,-1,-1,-1,-1)              % ECHO 1 - 9
106130   INTEGER ARRAY ECH4:=(-1,-1,77777,-1,100000,37,100000,37)       % ECHO LETTERS
106140   INTEGER ARRAY ECH5:=(-1,-1,0,0,0,0,0,3)                        % ECHO ALPHANUMERIC
106150   INTEGER ARRAY ECH6:=(177377,-1,77777,-1,100000,37,-1,-1)
106160   @ELIB
106160
106160
106160   %=============================================================================
106160   % 37.2        S T T I N   T Y E N T
106160   %
106160   % TERMINAL INPUT DRIVER, LEVEL 12
106160   %
106160   % THE X-REGISTER MUST POINT TO THE OUTPUT DATAFIELD IN ALL THE
106160   % ROUTINES CALLED BY THIS DRIVER
106160   %
106160   INTEGER 12DFOPP,10DFOPP,IISTATE,OISTATE
106164   SUBR STTIN,TYENT
106164   SYMBOL 5PIP=17
106164   @LIB CXCPU
106164   INTEGER POINTER 3PT12:=177600+L12LGP+L12LGP+1
106165   @ELIB
106165   STTIN:
106165   RETURN: TTMR=:TMR
106167           CALL ID12                                  % WAIT FOR INTERRUPT
106170   TYENT: ISTATE=:IISTATE
106172   @LIB CXCPU
106172           TDFPHPAGE=:3PT12; TDFLGADDR/\1777+"L12LGP*2000"=:B; *PON
106201           IF DFOPP><0 THEN X:=1777; X/\A; A:=B/\176000+X FI
106210           A=:12DFOPP=:X
106212   @ELIB
106212   @LIB CXCPU-,
```

```
106212          CALL IOAPD; GO RETURN                          % READ CHARACTER
106214          IF T:="ROUSPEC"><0 THEN
106217             D:=0; CALL ROUSPEC; GO RETURN; GO OKCHAR
106223          FI; A BZERO 7=:LAST                            % SAVE IN DATAFIELD
106225          CALL XONREAD; GO RETURN
106227  OKCHAR: BRECHOFL/\170=:BRECHOFL                        % CLEAR BREAK/ECHO FLAG
106232          A := LAST/\377; CALL XONCHECK                  % TEST FOR XON/XOFF OUT.
106235          IF DFLAG BIT 5CAPITAL THEN                     % CONVERT LOWER CASE TO UPPER CASE
106240             LAST/\177=:T
106243             IF 140<T AND 176>T AND CESCP SHZ -10><T AND CESCP/\377><T THEN
106261                LAST BZERO 5=:LAST                       % CONVERT ALL EXEPT
106264             FI                                          % RUBOUT  AND DISCONNECT
106264          FI
106264          IF X><0 AND X.SCREEN><0 THEN                   % TWO WAY DEVICE WITH CR-DELAY
106267                IF <0 THEN -1=:X.TMR FI
106272                1=:X.SCREEN
106274                IF LAST=0 THEN                           % IGNORE TAPE-FEED?
106276                  IF FLAGB NBIT 5LCHAR GO RETURN
106301                  IF CESCP SHZ-10><0 GO RETURN
106304                FI
106304          FI
106304          IF T:=DFLAG NBIT 5SPEC AND T NBIT 5CTRLO THEN
106311             CALL ESCAPE; GO RETURN; X:=12DFOPP
106314          FI
106314          IF DFLAG NBIT 5CTRLO THEN
106317             IF LAST/\377=17 THEN
106324                DFLAG BONE 5CTRLO=:DFLAG
106327             FI
106327          ELSE
106330             DFLAG BZERO 5CTRLO=:DFLAG
106333          FI
106333          IF CFREE=0 THEN
106335             TINFO BONE 5BFUL=:TINFO; GO BFULL           % BUFFER IS FULL, RESTART USER
106341          FI
106341          GO L1; *)FILL
106357  L1:    *IOF
106360          IF DFLAG BIT 5ECHO THEN
106363               LAST/\377
106365               CALL TECHO                  %TEST FOR ECHO
106366               CALL TBREAK                 %TEST FOR BREAK
106367               X:=12DFOPP
106370          ELSE
106371               IF IISTATE=-1 THEN; *ION    % NOWAIT MODE
106376                  LAST/\377; CALL CXRBPUT; GO BFULL
106402               FI
106402          FI
106402         *ION
106403          IF BRECHOFL BIT 5ECHO THEN CALL ECHSUBR FI              % ECHO CHAR?
106407          IF BRECHOFL<0 THEN 7=:LAST; CALL ECHSUBR; GO RETU FI    % IGNORE CHAR?
106415
106415  ADDCH: LAST/\377; CALL CXRBPUT                          % PUT CHARACTER IN BUFFER
106420
106420          IF BHOLD>=MAX-10 AND BRECHOFL NBIT 5BREAK GO BFULL % FULL BUFFER
106430          IF BRECHOFL BIT 5BREAK THEN                     % BREAK ACCORDING TO STRATEGY
106433             DFLAG BZERO 5ECHO                            % ECHO OFF IN DRIVER
106435             IF X:=FYLLE=HENTE THEN
106441                A BONE 5ALEC                              % WHOLE BUFFER IS ECHOED BY DRIVER
106442             ELSE
106443                A BZERO 5ALEC                             % PART OF BUFFER MUST BE ECHOED BY IOTRANS
106444             FI; A=:DFLAG; X=:RSISTE                      % RSISTE IS FIRST CHARACTER NOT ECHOED
```

```
106446          ELSE
106447             GO RETU
106450          FI
106450
106450    BFULL: CALL TSTBACK                                   % RESTART PROGRAM
106451    *"8N500
"106451           IF X:=IN5MSG >< 0 THEN
106453    @LIB CXCPU-,
106453    @LIB CXCPU
106453                T:=5MBBANK; *AAX MLFLA; LDATX
106456                IF A=0 THEN
106457                   A:=1; *STATX
106461                   "IBMOVE"; CALL CXRTACT
106463                FI
106463    @ELIB
106463          ELSE
106464    BYP1:         IF IISTATE=-1 OR A=-2 THEN; *IOF
106474                   CALL PNW5ST; GO NST; GO XRETU
106477    NST:           *ION
106500                FI
106500    @LIB CXCPU
106500                "IORESTART"; CALL CXRTACT
106502    @ELIB
106502    @LIB CXCPU-,
106502          FI
106502    *"
"106502     *"-8N500
"106502
106502    RETU:  IF "ROUSPEC"><0 THEN
106504              D:=1; X:=12DFOPP; CALL ROUSPEC; GO FAR OKCHAR
106510          FI
106510          GO FAR RETURN
106511    XRETU: *ION
106512          GO RETU
106513    *)FILL
106532    %
106532    %%  ECHO:
106532    INTEGER  LREG
106533    ECHSUBR: *IOF
106534          A:=L=: LREG
106536          IF X.CFREE=0 THEN
106540             *ION
106541             GO FAR RETURN
106542          FI
106542          IF X.MINBHOLD BIT 5BLOC THEN    % OUTPUT BUFFER LOCKED
106545             DFLAG BZERO 5ECHO=:DFLAG; BRECHOFL BZERO 5BREAK=:BRECHOFL
106553             -1=:RSISTE; *ION
106556             GO FAR ADDCH
106557          FI
106557          LAST/\377; X:=:B
106562          IF T:=EMPTFLAG><0 AND T:=X.DFLAG NBIT 5OXON THEN
106570             IF T:=TYPRING BIT 5CRDLY THEN CALL CHDL1 FI
106574             CALL ECAPD
106575          ELSE
106576    PIBUF:    CALL CXRBPUT
106577          FI
106577          X:=:B; LREG =: L
106602          *ION
106603          EXIT; *)FILL
106612
```

```
106617    F APL:  A=:D; T:=HDEV+DST; *IOXT                        % READ STATUS
106616            IF A BIT 3 THEN                                 % READY FOR TRANSFER
106620               D=:A; T-DST+DDW; *IOXT                       % WRITE CHARACTER
106624               "DACT"; T-DDW+DCONT; *IOXT                   % ACTIVATE
106630               EXIT
106631            ELSE                                            % NOT READY
106632               0=:EMPTFLAG; "DACT+DPIN"
106634               T-DST+DCONT; *IOXT                           % ACTIVATE AND PIN
106637               D=:A; GO PIBUF                               % ECHO VIA OUTPUT-BUFFER
106641            FI
106641            EXIT
106642    RBUS
106642    %
106642    %
106642    %=================================================================
106642    %37.2A        TECHO
106642    %
106642    %   TEST FOR ECHO:
106642    %   THIS SUBROUTINE SHOULD BE CALLED WITH INTERRUPTS OFF!!
106642    %
106642    SUBR TECHO
106642    INTEGER CHARN; INTEGER POINTER LREG
106644    SYMBOL 5PIP=17
106644    TECHO:  A=:CHARN; A:=L=:"LREG"
106647            IF X:=ECHOTAB><0 THEN
106651               A:=CHARN; CALL VSXGETBIT
106653               IF A<0 THEN                                  % NO ECHO
106654                  IF X:=ECHOTAB><"ECHO" AND X><"ECH1" AND X><"ECH2" THEN
106666                     BRECHOFL BONE 5PIP=:BRECHOFL
106671                  FI
106671               ELSE                                         % ECHO
106672                  IF DFLAG NBIT 5SPEC AND A NBIT 5HDUP THEN
106677                     BRECHOFL BONE 5ECHO BZERO 5PIP=:BRECHOFL
106703                  FI
106703               FI
106703            FI
106703           CHARN; GO LREG
106705    RBUS
106711    %=================================================================
106711    % 37.2B         TBREAK
106711    %
106711    %       TEST FOR BREAK:
106711    %       THIS ROUTINE SHOULD BE CALLED WITH INTERRUPTS OFF!!
106711    %
106711    SUBR TBREAK
106711    INTEGER CHARN; INTEGER POINTER LREG
106713    SYMBOL 5PIP=17
106713    TBREAK:  A=:CHARN; A:=L=:"LREG"
106716             IF DFLAG BIT 5SPEC THEN          %UNCONDITIONAL BREAK, NO ECHO.
106721                0=:NCBRK
106722                A BZERO 5SPEC BZERO 5CTRLO=:DFLAG
106725                BRECHOFL BONE 5BREAK BZERO 5ECHO=:BRECHOFL
106731             ELSE
106732                IF X:=BRKTAB><0 THEN
106734                   IF BRECHOFL>=0 THEN MIN NCBRK; 0/\0; FI
106740                   IF BRKTAB="BRK1" AND CHARN/\177=17 OR =20 OR =26 OR =30 OR =32 THEN
106765                      DFLAG BONE 5SPEC =:DFLAG      %BREAK ON NEXT
106770                   ELSE
106771                      A:=CHARN; CALL VSXGETBIT
106773                      IF A<0 THEN                    %BREAK ON CHARACTER
```

```
107774                      0=:NCBRK
107775                      BRECHOFL BONE 5BREAK BZERO 5PIP=:BRECHOFL
107001                   ELSE                          %POSSIBLE BREAK ON MAX CHARS.
107002                      IF BRKMAX><0 AND A<<=NCBRK THEN
107007                         0=:NCBRK
107010                         BRECHOFL BONE 5BREAK=:BRECHOFL
107013                      FI
107013                   FI
107013                FI
107013             FI
107013          FI; CHARN; GO LREG;  *)FILL
107020    RBUS
107020
107020    %============================================================================
107020    % 37.3        T T G E T     T D G E T
107020    %
107020    % TTGET:      TERMINAL IOTRANS ROUTINE. CALLED VIA DATAFIELD TO GET A
107020    %             CHARACTER FROM BUFFER.
107020    %             SKIP-RETURN : A = CHARACTER
107020    %             RETURN      : BUFFER EMPTY
107020    %                         : OR TERMINAL NOT CONNECTED (DERROR=316)
107020    %
107020    % TDGET:      TERMINAL IOTRANS ROUTINE 2. CALLED DIRECTLY.
107020    %             ROUTINE RETURNS CHARACTERS FROM BUFFER AND CHECKS FOR
107020    %             BREAK CONDITIONS IN DATA
107020    %             SKIP-RETURN : A = CHARACTER
107020    %                           T = 0: NO BREAK, T = 1: BREAK CONDITION
107020    %             RETURN      : BUFFER EMPTY
107020    %                         : OR TERMINAL NOT CONNECTED (DERROR=316)
107020    %
107020    SUBR TTGET,TDGET
107020    INTEGER CHARN,XREG,INPTY; INTEGER POINTER LREG
107024    INTEGER CDFOP
107025    TDGET: A:=0; GO FELLS
107027    TTGET: A:=1
107030    FELLS: A=:INPTY; X=:XREG; A:=L=:"LREG"
107034    @LIB CXCPU
107034           DFOPP/\1777=:X; A:=B/\176000; X+A
107042    @ELIB
107042    @LIB CXCPU-,
107042           X=:CDFOP
107043    NXTCH: BRECHOFL/\70=:BRECHOFL
107046           IF FLAGB BIT 5LSTA THEN                    % TERMINAL IS NOT CONNECTED
107051              TER02=:DERROR; GO OUT1
107054           FI
107054           IF BHOLD=0 THEN                            % EMPTY BUFFER
107056                 DFLAG BONE 5ECHO BZERO 5ALEC=:DFLAG % TURN ON ECHO IN DRIVER
107062                 -1=:RSISTE                          % DRIVER DEFINES NEXT BREAK
107064                 GO OUT1
107065           FI
107065           IF X.CFREE=0 THEN
107067              BRECHOFL BONE 5WECH=:BRECHOFL           % WAIT FOR OUTPUT BUFFER
107072              GO OUT1
107073           FI
107073           IF RSISTE>=0 THEN
107075              IF HENTE=RSISTE THEN -1=:RSISTE FI
107103           FI
107103           CALL XONCHECK; CALL CXRBGET; CALL SETPARITY; A=:CHARN
107107           IF DFLAG BIT 5ECHO GO RETU                 % ECHO BY DRIVER
107112           IF RSISTE<0 THEN                           % ECHO SHOULD BE DONE BY ME
```

```
107114              IF DFLAG NBIT 5ALEC THEN                % ALL CHARACTERS ALREADY ECHOED?
107117                 CHARN; CALL TECHO                    % TEST ECHO
107121              FI
107121              CALL TBREAK                             % TEST BREAK
107124              X:=CDFOP
107123              IF BRECHOFL<0 THEN            % CHARACTER SHOULD BE IGNORED
107125                 7=:CHARN; CALL GECHO; GO NXTCH
107131              FI
107131              IF A BIT 5ECHO THEN CALL GECHO FI       % GIVE ECHO
107134           FI
107134   RETU:   T:=0
107135           IF INPTY=0 THEN                  % BREAK CHECK IN TDGET
107137              IF RSISTE>=0 THEN
107141                 IF HENTE-RSISTE=0 THEN T:=1 FI      % BREAK BY DRIVER
107145              ELSE
107146                 IF BRECHOFL BIT 5BREAK THEN T:=1 FI % BREAK CONDITION
107152              FI
107152           FI
107152   OUT:    CHARN; MIN "LREG"
107154   OUT1:   X:=XREG; GO LREG
107156
107156   % GIVE ECHO FROM TTGET OR TDGET
107156   % B=INPUT DATAFIELD, X=OUTPUT DATAFIELD
107156   INTEGER POINTER LREGL
107157   GECHO:  A:=L=:"LREGL"
107161           X:=:B
107162           CHARN; CALL CXRBPUT; CALL STDEV; X:=:B
107166           GO LREGL
107167   RBUS
107203
107203   *"BXON
"107203   %=====================================================================
107203   %              X O F T R
107203   %       37.20
107203   %
107203   % THIS ROUTINE IS AN ALTERNATIVE TO NORMAL XOFF/XON CHECKING AND
107203   % REQUIRES XOFF/XON TO BE IN SEQUENCE. ANY OTHER COMBINATION WILL
107203   % BE TREATED AS NORMAL DATA CHARACTERS.
107203   % THE ROUTINE IS ACTIVATED BY ENTERING IT'S ADDRESS IN THE ROUSPEC
107203   % LOCATION IN THE TERMINAL INPUT DATAFIELD. THE ROUTINE IS CALLED
107203   % AT THE START (D=0) AND AT THE END (D=1) OF THE DRIVER.
107203   % FIRST PART (D=0) PERFORMS THE XOFF/XON CHECKING WILE THE LAST PART
107203   % (D=1) PERFORMS BRANCHING ACCORDING TO RESULTS FROM FIRST PART.
107203
107203   SUBR XOFTR
107203   SYMBOL IXOFF=3,IDATA=4,OPIN=5
107203   XOFTR:  IF T:=DFLAG BIT 5XDEVICE THEN   % XOFF/XON CHECKING FOR THIS TERMINAL?
107206              IF D=0 THEN                                % TEST CHARACTERS
107210                 A BZERO 7=:LAST=:D
107213                 IF T:=BRECHOFL NBIT IXOFF THEN      % X O F F   T E S T
107216                    IF "XOFF"=D THEN
107221                       T BONE IXOFF BZERO IDATA=:BRECHOFL
107224                       T:=DFLAG BONE 5OXON=:DFLAG            % STOP OUTPUT
107227                       0=:X.TMR; L-1
107231                    FI; A:=LAST; EXITA
107233                 ELSE                                % X O N   T E S T
107234                    IF D="XON" THEN                    % IGNORE CHARACTERS
107237                       T:=BRECHOFL BZERO IXOFF BZERO IDATA=:BRECHOFL
107243                       T:=DFLAG BZERO 5OXON=:DFLAG
107246                       T:=X.HDEV+DCONT
```

```
107250                      A:=1; *IOXT                                 % PIN OUTPUT
107252                      EXIT
107253                   ELSE
107254                      A SHZ 10+"XOFF"=:LAST                       % DATA CHARACTERS
107257                      T:=BRECHOFL BONE IDATA BZERO IXOFF=:BRECHOFL
107263                      EXITA
107264                   FI
107264                FI
107264             ELSE                                  % R E T U R N   C H E C K
107265                IF T:=BRECHOFL BIT IDATA THEN
107270                   T BONE OPIN BZERO IDATA=:BRECHOFL
107273                   LAST SHZ -10=:LAST
107276                   EXIT                                           % NEW CHARACTER
107277                FI
107277                IF T:=BRECHOFL BIT OPIN THEN
107302                   T BZERO OPIN=:BRECHOFL
107304                   DFLAG BZERO 50XON=:DFLAG
107307                   T:=X.HDEV+DCONT
107311                   A:=1; *IOXT                                    % PIN OUTPUT
107313                FI
107313                EXITA
107314             FI
107314          ELSE
107315             IF D=0 THEN
107317                A BZERO 7=:LAST; 0=:BRECHOFL
107322             FI
107322          FI
107322          EXITA
107323    RBUS
107323    *"
```

```
"107323
107323
107323    %=============================================================================
107323    % 37.4        D W R I T E
107323    %
107323    % DRIVER ROUTINE ON LEVEL 10
107323    %
107323    % THE X-REGISTER MUST POINT TO THE INPUT DATAFIELD (X=DFOPP)
107323    % IN ALL THE ROUTINES CALLED BY THE OUTPUT DRIVER
107323
107323
107323    %
107323
107323    INTEGER LOGFIELD=?
107323    INTEGER 10BUFST
107324    SUBR DWRITE
107324    SYMBOL BIT0=0
107324    SYMBOL SUFFS=5                  % SUFFICIENT SPACE FOR ECHO
107324    INTEGER CCNOBYT,XCNOBYT
107326    INTEGER POINTER 3PT10:=177600+L10LGP+L10LGP+1
107327    DWRITE:
107327            ISTATE=:OISTATE
107331    @LIB CXCPU
107331            IF TYPRING BIT 5TERM THEN
107334                TDFPHPAGE=:3PT10; TDFLGADDR/\1777+"L10LGP*2000"=:B; *PON
107343                IF DFOPP><0 THEN X:=1777; X/\A; A:=B/\176000+X FI
107352                A=:X; A:=B/\176000+BUFST=:10BUFST
107357            ELSE X:=DFOPP
107361            FI; X=:10DFOPP
107362    @ELIB
107362    @LIB CXCPU-,
107362            *IOF
107363    *"8N500
"107363             IF X><0 AND X.TYPRING BIT 5TERM AND X:=ON5MSG><0 THEN
107371               T:=5MBBANK; *AAX NOBYT; LDATX; STA XCNOB
107375               IF BHOLD = 0 THEN 0=:FYLLE=:HENTE; GO MOVBUF FI
107402               IF FYLLE NBIT 0 AND CFREE BZERO BIT0 >= XCNOBYT THEN
107412    MOVBUF:       IF XCNOBYT > CFREE BZERO BIT0 THEN
107417                      IF A>T+T THEN T SHZ -1; A:=T ELSE T SHZ -1; A-T+1 BZERO "0" FI
107431                   FI; T:=CFREE-A=:CFREE
107434                   A=:CCNOBYT+BHOLD=:BHOLD:=XCNOBYT-CCNOBYT=:XCNOBYT
107442                   T:=5MBBANK; *STATX
107444                   FYLLE SHZ -1+10BUFST=:L
107450                   IF FYLLE+CCNOBYT >= MAX THEN A-T FI; A=:FYLLE
107457                   MAX SHZ -1+10BUFST=:D
107463                   T:=5MBBANK; *AAX HBUFA-NOBYT; LDATX
107466                   *AAX 5HENT-HBUFA; LDXTX; COPY SA DT
107471                   *ION
107472                   DO WHILE CCNOBYT > 0
107475                      A-2=:CCNOBYT
107477                      * LDATX
107500                      X+1; X:=:L; IF X = D THEN X:=10BUFST FI
107505                      A=:X.S0; X+1; X:=:L
107510                   OD; *IOF
107512                   X=:A:=ON5MSG; T:=5MBBANK; *AAX 5HENT; STATX; AAX -5HENT
107520                   IF XCNOBYT<=0 THEN 0=:ON5MSG; CALL PT5RST FI
107525
107525               FI
107525            FI; X:=10DFOPP; GO INDRIV; *)FILL
107536    *"
```

```
"107536   INDRIV:
107536          MINBHOLD BZERO 5BLOC=:T
107541          IF BHOLD<T AND OISTAT><0 THEN
107546   @LIB CXCPU-,
107546   @LIB CXCPU
107546             IF TYPRING BIT 5TERM THEN "IORESTART"; CALL CXXRTACT ELSE CALL XRTACT FI
107555             X:=10DFOPP
107556   @ELIB
107556          FI
107556          IF SCREEN><0 THEN
107560             IF A>24 THEN 207; CALL OOAPD; 0=:EMPTFLAG; -1=:SCREEN FI
107570             IF A<0 GO FAR OUT
107572          FI; CALL OXONCHECK; GO CALLOOAPD          % SEND 'XON/XOFF'
107574          CALL XONWRITE; GO FAR OUT                 % TEST FOR XOFF.
107576   NOXON: IF BHOLD >< 0 THEN
107600             T:=HDEV+DST; *IOXT
107603             IF BIT 3 THEN                    % READY FOR TRANSFER AND CARRIER NOT MISSING
107605                IF TYPRING BIT 5CRDLY THEN
107610                   CALL CHDL2
107611                ELSE
107612                   IF A BIT 5TERM THEN CALL CXRBGET ELSE CALL RBGET FI
107617                FI
107617   NOCH:        IF T:=SCREEN>0 THEN
107622                   IF =12 THEN MIN SCREEN FI
107626                   IF =14 THEN T+24=:SCREEN FI
107633                   GO CALLOOAPD; *)FILL
107651                FI
107651   CALLOOAPD:   CALL XOOAPD; 0=:EMPTFLAG; TTMR=:TMR
107655                IF X><0 THEN X.DFLAG BZERO 5LBRK=:X.DFLAG FI  % LINE IS OK
107661             ELSE
107662                T:=HDEV+"DCONT"; "DPIN+DACT"; *IOXT           % PIN DEVICE.
107666                TTMR =: TMR                                   % START TIME-OUT
107670                IF X><0 THEN
107671                   IF X.TYPRING BIT 5TERM THEN                % SPEC. FOR TERMINALS
107674                      T := X.HDEV + DST; * IOXT               % READ INPUT STATUS.
107677                      IF BIT 4 OR BIT 13 THEN                 % IF ERROR
107703                         IF BIT 13 THEN                       % CARR. MISSING
107705                            X.DFLAG BONE 5LBRK=:X.DFLAG       % FLAG CARRIER MISSING
107710                            -2=:X.TMR                         % SET INPUT TIME-OUT IN 2 SEC.
107712                         FI; T:=HDEV+"DCONT"; "DACT"; *IOXT   % UNPIN DEVICE
107716                         CALL CLBUF                           % CLEAR BUFFER
107717   @LIB CXCPU
107717                         IF TDRADDR.RTRES><0 THEN
107722                            X:=:B; CALL XRTACT; B:=X
107725                         FI; X:=10DFOPP
107726   @ELIB
107726   @LIB CXCPU-,
107726   EFI:        FI; FI; FI; FI
107726          ELSE
107727   *"8N500
"107727              GO IN500; *)FILL
107734   INTEGER CNOBYT,YCNOBYT
107736   IN500:      IF X><0 AND X.TYPRING BIT 5TERM AND X:=ON5MSG >< 0 THEN
107744                   MAX /\ 177774=:MAX; T:=5MBBANK; *AAX NOBYT; LDATX; STA YCNOB
107753                   IF A > MAX THEN
107756                      IF A>T+T THEN MAX ELSE A-MAX FI
107764                   FI; T:=CFREE-A=:CFREE
107767                   A=:CNOBYT+BHOLD=:BHOLD:=YCNOBYT-CNOBYT=:YCNOBYT; T:=5MBBANK; *STATX
107777                   10BUFST=:L; *AAX HBUFA-NOBYT; LDATX
110003                   *AAX 5HENT-HBUFA; LDXTX; COPY SA DT; ION
```

```
110007                  DO WHILE CNOBYT > 0
110012                     A-4=:CNOBYT
110014                     * LDDTX
110015                     X+2; X:=:L; AD=:X.DSO; X+2; X:=:L
110022                  OD; 0=:FYLLE=:HENTE; *IOF
110026                  X=:A:=ON5MSG; T:=5MBBANK; *AAX 5HENT; STATX; AAX -5HENT
110034                  IF YCNOBYT <= 0 THEN 0=:ON5MSG; CALL PT5RST FI
110041                  X:=10DFOPP; GO FAR INDRIV
110043               FI
110043   *"8LOG
"110043   @LIB CXCPU
110043               IF TYPRING NBIT 5TERM GO NLOG1
110046               *POF; LDA I (LOGFI; PON
110051               IF A=TDRADDR THEN
110054   @ELIB
110054   @LIB CXCPU-,
110054                  0=:EMPTFLAG; A:=0; CALL OOAPD; TTMR=:TMR
110061               ELSE
110062   *"
"110062    NLOG1:        1=:EMPTFLAG; 0=:TMR
110065               FI
110065            FI;  GO OUT; *)FILL
110076
110076   INTEGER SAVB
110077
110077   OUT:    IF CFREE=SUFFS THEN              % CHECK IF INPUT IS WAITING FOR BUFFER
110103              IF X:=10DFOPP><0 THEN
110105   @LIB CXCPU
110105                 IF X.BRECHOFL BIT 5WECH THEN
110110                    A BZERO 5WECH=:X.BRECHOFL; X:=:B=:SAVB
110114   *"8N500
"110114                    IF X:=IN5MSG >< 0 THEN
110116                       T:=5MBBANK; A:=1; *AAX MLFLA; STATX
110122                       "IBMOVE"
110123                    ELSE
110124                       CALL TSTBACK; "IORESTART"
110126                    FI
110126   *"-8N500
"110126                    CALL CXXRTACT; SAVB=:B
110131                 FI
110131   @ELIB
110131   @LIB CXCPU-,
110131               FI
110131            FI; *ION
110132   @LIB CXCPU-,
110132   @LIB CXCPU
110132            IF TYPRING NBIT 5TERM GO NLOG2
110135            X:=TDRADDR
110136            *POF; JPL I (LOG1; PON
110141   @ELIB
110141   NLOG2: CALL ID10; GO FAR DWRITE
110143
110143   RBUS
110154
110154
110154   %=============================================================================
110154   % 37.5        I O A P D   O O A P D
110154   %
110154   % SUBROUTINES TO READ AND WRITE A CHARACTER USING IOXT!!!!!!!
110154
```

```
110154    SUBR IOAPD,OOAPD,XOOAPD
110154    INTEGER ERRCODE,IOXNO
110156    IOAPD: T:=HDEV+DST; *IOXT
110161           A =: D                                    % STATUS
110162           T+"DDR-DST"; *IOXT                        % BYTE IN A
110164           IF D BIT 13 THEN                          % CARRIER MISSING??
110166                IF TYPRING BIT 5TERM THEN            % TERMINAL
110171                   DFLAG BONE 5LBRK=:DFLAG
110174                   -2=:TMR                           % START TIME-OUT IN 2 SEC.
110176                FI
110176                L =: D                               % SAVE RETUR-ADRESS
110177                CALL CLBUF ; L := D                  % CLEAR BUFFER
110201                EXIT
110202           ELSE
110203              T:=DFLAG BZERO 5LBRK=:DFLAG           % LINE IS OK
110206           FI
110206           IF D BIT 4 THEN
110210              T:=TINFO
110211              IF D BIT 5 THEN T BONE 5FRER ELSE T BONE 5PAER FI; T=:TINFO
110217              IF A=0 THEN
110220                 IF CESCP/\377><LAST GO SIMESC
110225              FI
110225              CNTREG; T:=HDEV+DCONT; *IOXT
110231              TTMR=:TMR; EXIT
110234           FI
110234    SIMESC: A=:D; CNTREG; T:=HDEV+DCONT; *IOXT
110241           D=:A; EXITA
110243
110243    OOAPD: A=:D                                      % SAVE CHAR IN D
110244           T:=HDEV+DST; *IOXT
110247           IF BIT 3 THEN                             % READY!!!
110251              D=:A
110252    XOOAPD:   T:=HDEV+DDW; *IOXT                     % WRITE CHAR TO INTERFACE
110255           FI
110255    OUTRA: "DACT+DPIN"; T:=HDEV+"DCONT"; *IOXT       % RE-ACTIVATE.
110261           EXIT
110262
110262    RBUS
110264
110264
110264    @LIB CXCPU
110264    %=============================================================================
110264    %              C X R B P U T   -   C X R B G E T
110264    %
110264    % IOTRANS ROUTINES FOR TERMINALS
110264    %
110264    SUBR CXRBPUT,CXRBGET
110264    DISP 0; REAL FFMAX=MAX; DOUBLE DBHOLD=BHOLD; PSID
110264    CXRBPUT: IF T:=CFREE=0 THEN EXIT FI
110270             T-1=:CFREE; X=:D:=FYLLE; A=:T:=B/\1776000+BUFST; A:=:T; *SBYT
110302             X+1; IF X=MAX THEN X:=0 FI; X=:FYLLE
110310             MIN BHOLD; X:=D; EXIT
110313
110313    CXRBGET: TAD:=FFMAX; IF A=0 THEN EXIT FI
110316             A-1; D+1; AD=:DBHOLD
110321             IF D=T THEN 0=:HENTE FI; D-1:=:X
110326             A:=B/\176000+BUFST=:T; *LBYT
110333             MIN CFREE; X:=D; EXIT
110336    RBUS
110337
```

```
110337    !LIB
110337    !=============================================================================
110337    ! 37.7        R B G E T   R B P U T     R W G E T   R W P U T   D B P U T
110337    !   - - -3.5.6.2  A U X I L I A R Y     R O U T I N E S
110337
110337    ! RING BUFFER ROUTINES TO GET OR PUT A BYTE
110337    ! CHARACTER IN A
110337    !       DBPUT PUTS A BYTE INTO A DEVICE BUFFER SIMULATING A RING BUFFER.
110337    !             - IT SHOULD BE EXECUTED WITH PIOF AND ON ENTRY A SHOULD HAVE
110337    !             - THE BYTE TO BE STORED IN ITS LEAST SIG. BYTE.
110337    !
110337    ! TDBGET/TDBPUT: T=MEMORY BANK OF BUFFER
110337    !                ELSE SAME FUNCTION AS DBGET/DBPUT
110337    !
110337    !LIBR RBGET,RBPUT,RWGET,RWPUT,DBPUT,DBGET,TDBPUT,TDBGET
110337    !
110337    INTEGER TREG, AREG, DREG, XREG          % SAVE REGISTERS FOR DBGET
110343    REAL TADREG=TREG
110343    INTEGER POINTER LREG
110344    DISP 0; REAL FFMAX=MAX; DOUBLE DBHOLD=BHOLD
110344            INTEGER ARRAY POINTER RWBUFST=BUFST
110344    PSID
110344    RBGET:  TAD:=FFMAX; IF A=0 THEN EXIT FI     %BHOLD
110347            A-1; D+1; AD=:DBHOLD
110352            IF D=T THEN 0=:HENTE FI      % BOTTOM OF BUFFER
110355            D-1:=:X; T:=BUFST; *LBYT
110361            MIN CFREE; D=:X; EXIT
110364
110364    RBPUT:  IF T:=CFREE=0 THEN EXIT FI; T-1=:CFREE
110372            X:=:D:=FYLLE; T:=BUFST; *SBYT
110376            X+1; IF X=MAX THEN X:=0 FI; X=:FYLLE
110404            MIN BHOLD; D=:X; EXIT
110407
110407    INTEGER CMBBANK
110410
110410    TDBPUT: TAD=:TADREG; GO DDBPU
110412    DBPUT:  TAD=:TADREG; T:=BUFBANK
110414    DDBPU:  T=:CMBBANK
110415            X=:XREG; A/\377=:D
110420            IF CFREE><0 THEN
110422               A-1=:CFREE; MIN BHOLD
110425               X:=BUFST; A:=FYLLE; *BLDA 00 DA
110430               A SHZ -1; X+A; T:=CMBBANK; *LDATX
110434               IF K THEN A/\177400 ELSE D SHZ 10; A/\377 FI
110442               A\/D; *STATX
110444               IF A:=FYLLE+1=MAX THEN A:=0 FI; A=:FYLLE
110453            FI
110453    OUT:    TAD:=TADREG; X:=XREG; EXIT
110456
110456    TDBGET: TAD=:TADREG; GO DDBGE
110460    DBGET:  TAD=:TADREG; T:=BUFBANK
110462    DDBGE:  T=:CMBBANK
110463            X=:XREG; TAD:=FFMAX
110465            IF A><0 THEN                                  % BHOLD
110466               A-1; D+1; AD=:DBHOLD; IF D=T THEN 0=:HENTE FI
110474               D-1; *BLDA 00 DD
110476               D SHZ -1; X:=BUFST+D; T:=CMBBANK; *LDATX
110503               IF K NBIT THEN A SHZ -10 FI; A/\377=:AREG
110510               MIN CFREE
110511            FI; GO OUT
```

```
110512
110512    *)FILL
110515
110515    RWGET: IAD:=FFMAX; IF A=0 THEN EXIT FI     %BHOLD
110520           A-1; D+1; AD=:DBHOLD; IF D=T THEN 0=:HENTE FI
110526           D-1:=:X; RWBUFST(X); MIN CFREE; D=:X; EXIT
110534
110534    RWPUT: IF T:=CFREE=0 THEN EXIT FI; T-1=:CFREE
110542           X=:D; A=:RWBUFST(FYLLE)
110545           X+1; IF X=MAX THEN X:=0 FI; X=:FYLLE
110553           MIN BHOLD; D=:X; EXIT
110556    RBUS
110556    *"-CXCPU
"110556
110556    %=============================================================================
110556    % 40.1        T E L I N   T E L E N T
110556    %
110556    *"TELIX
"110556    %
110556    % SIMPLE DRIVER FOR TELETYPE INTERFACE
110556    % BREAK ON EVERY CHARACTER, NO ECHO, NO MODIFICATIONS OF THE BYTES
110556    % LEVEL 12
110556    %
110556
110556    SUBR TELIN,TELENT
110556    INTEGER ERRCODE,IOXNO
110560    @LIB CXCPU
110560    INTEGER POINTER 3PT12:=177600+L12LGP+L12LGP+1
110561    @ELIB
110561    TELIN: TTMR=:TMR;CALL ID12
110564    TELENT:
110564    @LIB CXCPU
110564           IF TYPRING BIT 5TERM THEN
110567              TDFPHPAGE=:3PT12; TDFLGADDR/\1777+"L12LGP*2000"=:B; *PON
110576              IF DFOPP><0 THEN X:=1777; X/\A; A:=B/\176000+X FI
110605              A=:X
110606           FI
110606    @ELIB
110606    @LIB CXCPU-,
110606           T:=HDEV+"DST"; *IOXT              % READ STATUS
110611           IF A BIT 4 THEN        % ERROR
110613                 A=:ERRCODE; A BONE 17=:DERROR; T-"DST"=:IOXNO; T+"DDR"; *IOXT
110622                 A=:D; T+"DST-DDR"; *IOXT
110625                 IF A BIT 3 GO ERR; GO L1
110630           ELSE
110631           T+"DDR-DST"; *IOXT     % READ DATA
110633           FI; A=:LAST; T+"DST-DDR"; *IOXT
110636           IF A BIT 3 GO ERR
110640    L1:    CNTREG; T+"DCONT-DST"; *IOXT      % PIN DEV
110643           A:=LAST; CALL XONCHECK
110645           CALL XONREAD; GO CONT1
110647           A :=  LAST
110650    % TEXT EDITING TERMINAL CONVERTING ROUTINE
110650    * TELIA 7TEXT
"110650     CONT1:
110650    @LIB CXCPU
110650           IF T:=TYPRING BIT 5TERM THEN
110653              CALL CXRBPUT; TDRADDR.MFUNC; CALL CXRTACT
110657           ELSE
110660              CALL RBPUT; CALL RTACT
```

```
110660            FI
110662    $ELIB
110662    $LIB CXCPU-,
110662            GO TELIN
110663    ERR:    T:=HDEV; A:=:T; CALL 9ERR(#23); 232=:DERROR
110671    $LIB CXCPU
110671            IF TYPRING BIT 5TERM THEN
110674               IF TDRADDR.RTRES><0 THEN X.MFUNC; CALL CXRTACT FI
110701            ELSE
110702               IF RTRES><0 THEN CALL RTACT FI
110705            FI
110705    $ELIB
110705    $LIB CXCPU-,
110705            CALL ID12; GO ERR22
110707    RBUS
```

```
110725
110725    *7TEXT
"110725   ==========================================================================
110725    40.2        C H D L 1    C H D L 2
110725

)9SCLC
110725    %
110725    % ROUTINES FOR GIVING 5 DUMMY CHARACTERS AFTER EACH CARRIAGE-RETTURN
110725    % THE DUMMY CHARATER IS 26 OCTAL
110725    %
110725
110725    % CHDL1 IS CALLED  FROM THE ROUTINE ECHSUBR
110725    %
110725    % ENTRY: A=CHARACTER TO OUTPUT (ECHO)
110725    %
110725    % EXIT:  A=OLD CHARACTER OR A=26 (DUMMY CHARACTER)
110725
110725    CHDL1, COPY SL DT; STT CHDLL; COPY SA DD; JPL CHDLF; JMP CHDLA
110732           COPY SD DA; LDT TYPRI,B; BSKP ONE 50 DA; JMP NTRM2  % TEST ON BIT 5TERM
110736           JPL I (CXRBP; SAA 26; JMP I CHDLL
110741    NTRM2, JPL I (RBPUT; SAA 26; JMP I CHDLL
110744
110744    CHDLA, COPY SD DA; JPL XXCRT
110746    CHDLD, COPY SD DA; JMP I CHDLL
110750
110750    % TEST IF CURRENT CHARACTER IS CR AND SET "DUMMY-CHARACTER" COUNTER TO 5
110750    % WHEN CHAR IS CR (THE COUNTER IS BIT 6-8 IN TYPRING)
110750    %
110750    XXCRT, AND (177; AAA -15; JAZ *+2; EXIT
110754           LDA (500; ORA TYPRI,B; STA TYPRI,B; EXIT
110760
110760    % TEST IF MORE DUMMY CHARACTERS MUST BE GIVEN
110760    % EXIT:       NO MORE DUMMY CHARACTERS
110760    % EXIT+1:     GIVE DUMMY CHARACTER
110760    %
110760    CHDLF, LDA TYPRI,B
110761           SHA ZIN SHR 6; AND (7; JAF *+2; EXIT
110765           AAA -1; SHA 6; LDT TYPRI,B; SWAP SA DT; AND (177077
110772           RORA ST DA; STA TYPRI,B; EXIT AD1
110775
110775    % ENTRY FROM THE ROUTINE DWRITE
110775    % TEST IF THE CHARACTER TO OUTPUT SHOULD BE FETCHED FROM THE RING-BUFFER
110775    % OR A DUMMY CHARACTER SHOULD BE GIVEN.
110775    % EXIT WITH THE CHARACTER TO OUTPUT IN A-REG
110775    %
110775    CHDL2, COPY SL DA; STA CHDLL; JPL CHDLF; JMP CHDLC; SAA 26
111002    CHDLB, JMP I CHDLL
111003    CHDLC, LDT TYPRI,B; BSKP ONE 50 DT; JMP NTRM1               % TEST ON BIT 5TERM
111006           JPL I (CXRBG; JMP FELLS
111010    NTRM1, JPL I (RBGET
111011    FELLS, COPY SA DD; JPL XXCRT; COPY SD DA; JMP CHDLB
111015
111015    CHDLL, 0
111016
111016    )FILL
111026    )KILL CHDLA XXCRT CHDLF CHDLB CHDLC CHDLL CHDLD NTRM2 NTRM1 FELLS
111026
111026    )9RCLC
)9SLPL 111026    %=========================================================================
```

```
 11026    % 40.5        X O N C H E C K   X O F F C H E C K   O X O N C H E C K
*11026    %
*11026    % SUBROUTINE TO TELETYPE DRIVERS TO CONTROL
*11026    % XON/XOFF SENDING TO TELETYPE INTERFACES
*11026    % XOFF IS GIVEN WHEN ONLY 5 FREE CHAR-ENTRIES IN BUFFER
*11026    % XON IS GIVEN WHEN HALF THE BUFFER IS FREE AND "XOFF" HAS
*11026    % PREVIOUSLY BEEN GIVEN
*11026    % LEVEL 12
*11026    %
*11026    SUBR XONCHECK,OXONCHECK
*11026    INTEGER POINTER LREGL
*11027    XONCHECK:
*11027            IF X=0 OR DFLAG NBIT 5RDEVICE THEN EXIT FI
*11034            A:=L=:"LREGL"
*11036            IF DFLAG BIT 5XOFF OR A BIT 6XOFF GO CONT1
*11043            IF BHOLD>MAX-5 THEN
*11050                  IF X.EMPTFLAG><0 THEN
*11052                     0=:X.EMPTFLAG; "XOFF"; X:=:B; CALL OOAPD; X:=:B
 11057                  ELSE
 11060                    DFLAG BONE 5XOFF=:DFLAG
 11063                  FI
 11063                  DFLAG BONE 6XOFF=:DFLAG
 11066                  GO CONT2
 11067            FI
 11067    CONT1:  IF DFLAG BIT 5XON OR A NBIT 6XOFF GO CONT2
 11074            IF CFREE=MAX SHZ -1 THEN
 11101                  IF X.EMPTFLAG><0 THEN
 11103                     0=:X.EMPTFLAG; "XON"; X:=:B; CALL OOAPD; X:=:B
 11110                  ELSE
 11111                    DFLAG BONE 5XON=:DFLAG
 11114                FI; DFLAG BZERO 6XOFF=:DFLAG
 11117            FI
 11117    CONT2:  GO LREGL
 11120
 11120    % OUTPUT LEVEL, (LEVEL 10)
 11120    OXONCHECK:
 11120            IF X=0 OR T:=X.DFLAG NBIT 5RDEVICE THEN EXITA FI
 11125            IF T:=X.DFLAG BIT 5XON THEN "XON"; T BZERO 5XON=:X.DFLAG; EXIT FI
 11134            IF T BIT 5XOFF THEN "XOFF"; T BZERO 5XOFF=:X.DFLAG; EXIT FI
 11142            EXITA
 11143
 11143    RBUS
 11144
 11144    *"-8RON
```

```
"111144
111144    * 8XON
"111144    %=============================================================================
111144    % 40.6          X O N R E A D
111144    %
111144    %
111144    %       SUBROUTINE TO INPUT DRIVERS
111144    %       TO CHECK FOR RECIEVED XON/XOFF FROM LEGAL DEVICE
111144    %       LEVEL 12.
111144    %
111144    %       EXIT  => IGNORE THIS CHAR.
111144    %       EXITA => CHAR. OK AS INPUT.
111144    %
111144    %
111144    %
111144    SUBR XONREAD
111144    XONREAD:  A/\177=:D                          % SAVE CHAR FOR TESTS & RETUR.
111146          IF DFLAG NBIT 5XDEVICE GO OKUT         % NOT ALLOWED
111151          IF D><"XON" AND D><"XOFF" GO OKUT      % NOT XON/XOFF
111157          IF D="XON" THEN                        % XON RECIEVED.
111162                A BZERO 5OXON =: DFLAG           % FLAG XOFF-STATE
111164                T := X.HDEV + DCONT              % RESTART OUTPUT DRIVER
111166                A:=1; *IOXT
111170          ELSE
111171                A BONE 5OXON =: DFLAG            % XOFF; FLAG.
111173                0=:X.TMR
111174          FI
111174          EXIT                                   % IGNORE-RETURN.
111175
111175    OKUT:   EXITA                                % OK RETURN
111176
111176    RBUS
111177
111177    %=============================================================================
111177    % 40.7          X O N W R I T E
111177    %
111177    %       ROUTINE TO CHECK FOR XOFF-STATE
111177    %       OUTPUT LEVEL (LEVEL 10)
111177    %
111177    %       EXIT  => XOFF STATE / DON'T SEND ANYTHING.
111177    %       EXITA => XON STATE / SEND
111177    %
111177    %             CALLED IN IOF
111177    %             =============
111177    %
111177
111177    SUBR XONWRITE
111177
111177    XONWRITE:
111177          IF X=0 OR X.DFLAG NBIT 5XDEVICE THEN EXITA FI
111204          IF A NBIT 5OXON THEN EXITA FI
111207          EXIT
111210
111210    RBUS
111210    * -8XON
"111210
111210    * 18 EXCPU
111210
111210    SUBR M88IWORD,B8IWORD,M8IDRET
111210    RBUS
```

```
111210
111210    % ------------------------------------------------------------------------
111210    %   B 8 0 T E R M   -   M 8 0 T E R M   -   M 8 0 I N D V   -   B 8 0 I N D V
111210    %
111210    $ BR B80TERM,M80TERM,M80INDV,B80INDV
111210
111210    INTEGER CBHOLD,CCHAR
111212
111212    CHCH1: A SHZ -10
111213    CHCHA: A/\377
111214           IF K THEN
111216              IF A=0 THEN CBHOLD; GO GORET FI
111221              MIN CBHOLD
111222           FI; A=:CCHAR; *SBYT
111224           IF MAX=X+1 THEN X:=0 FI
111231           EXIT
111232
111232    M80TERM: K:=1; GO BTRFEL
111234    B80TERM: K:="0"
111235    BTRFELL: CALL SETBFPAGE; *PON
111237           IF CFREE<10 GO TERWDX
111243           T:=BUFST; A:=B/\176000; T\/A
111247    FELLS: 0=:CBHOLD; X:=FYLLE
111251           *IRR ALEVB DA; JPL CHCH1
111253           *IRR ALEVB DA; JPL CHCHA
111255           *IRR ALEVB DD; JPL CHCH1
111257           *IRR ALEVB DD; JPL CHCHA
111261           *IRR ALEVB DL; JPL CHCH1
111263           *IRR ALEVB DL; JPL CHCHA
111265           *IRR ALEVB DX; JPL CHCH1
111267           *IRR ALEVB DX; JPL CHCHA
111271           A:=10
111272    GORET: A=:D+BHOLD=:BHOLD:=CFREE-D=:CFREE; X=:FYLLE
111301           *IRR ALEVB DT
111302           IF A>177 AND A<300 THEN
111310              CCHAR=:CHARI; DFOPP=:B
111314           FI; GO M8RET
111315
111315    M80INDV: K:="1"
111316           IF "IOTRANS"><"IPTCH" GO M88IWORD
111322    INDFEL: IF DFOPP.CFREE<10 GO WDX
111327           X=:B; T:=BUFST; GO FELLS
111332
111332    B80INDV: IF "IOTRANS"><"IPTCH" GO B8IWORD
111336             K:="0"; GO INDFEL
111340
111340    RBUS
111352
111352    %========================================================================
111352    %       M 8 I N T E R M
111352
111352    SUBR M8INTERM
111352    INTEGER CCHAR
111353    M8INTERM:
111353           IF TYPRING BIT 5TERM THEN
111356              CALL SETBFPAGE; *PON
111360           FI; CALL IOTRANS; GO OUT; A SHZ 10=:CCHAR; *IRW ALEVB DA;
111365           *ION; IOF
111367           X:=1; CALL IOTRANS; GO GORET; A\/CCHAR; *IRW ALEVB DA;
111374           *ION; IOF
```

```
111376          X+1; CALL IOTRANS; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DD;
111404          *ION; IOF
111406          X+1; CALL IOTRANS; GO GORET; A \/CCHAR; *IRW ALEVB DD;
111413          *ION; IOF
111415          X+1; CALL IOTRANS; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DL;
111423          *ION; IOF
111425          X+1; CALL IOTRANS; GO GORET; A\/CCHAR; *IRW ALEVB DL;
111432          *ION; IOF
111434          X+1; CALL IOTRANS; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DX;
111442          *ION; IOF
111444          X+1; CALL IOTRANS; GO GORET; A\/CCHAR; *IRW ALEVB DX;
111451          X+1
111452  GORET:  A:=X; *IRW ALEVB DT
111454          GO M8RET
111455  OUT:    IF TYPRING BIT 5TERM GO TERWDX
111460          GO WDX
111461  RBUS
111465
111465  %=======================================================================
111465  %       M 8 I B T E R M
111465
111465  SUBR M8IBTERM
111465  INTEGER CCHAR
111466  M8IBTERM: CALL SETBFPAGE; *PON
111470          CALL TDGET; GO TERWDX; A SHZ 10=:CCHAR; *IRW ALEVB DA;
111475          X:=1; IF T><0 GO BBREK; *ION; IOF
111502          CALL TDGET; GO GORET; A\/CCHAR; *IRW ALEVB DA;
111506          X+1; IF T><0 GO BBREK; *ION; IOF
111513          CALL TDGET; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DD;
111520          X+1; IF T><0 GO BBREK; *ION; IOF
111525          CALL TDGET; GO GORET; A \/CCHAR; *IRW ALEVB DD;
111531          X+1; IF T><0 GO BBREK; *ION; IOF
111536          CALL TDGET; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DL;
111543          X+1; IF T><0 GO BBREK; *ION; IOF
111550          CALL TDGET; GO GORET; A\/CCHAR; *IRW ALEVB DL;
111554          X+1; IF T><0 GO BBREK; *ION; IOF
111561          CALL TDGET; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DX;
111566          X+1; IF T><0 GO BBREK; *ION; IOF
111573          CALL TDGET; GO GORET; A\/CCHAR; *IRW ALEVB DX;
111577          X+1; IF T><0; GO BBREK; *ION; IOF
111604  GORET:  A:=X; *IRW ALEVB DT
111606          GO M8RET
111607  BBREK:  A:=X BONE 17; *IRW ALEVB DT
111612          GO M8RET
111613  RBUS
111617
111617  %=======================================================================
111617  %       B 8 I N T E R M   -   I B 8 I N T E R M
111617
111617  SUBR B8INTERM,IB8INTERM
111617  INTEGER CCHAR
111620  IB8INTERM: T:=BUFST; BHOLD=:D; GO FELLS
111624  B8INTERM: CALL SETBFPAGE; *PON
111626          IF FLAGB BIT 5LSTA THEN TER02=:DERROR; GO TERWDX FI
111634          IF BHOLD=0 GO TERWDX
111637          A=:D; A:=B/\176000+BUFST=:T
111644  FELLS:  MAX=:L; X:=HENTE; *LBYT
111650          A SHZ 10
111651          IF D-1=0 THEN
111654                  *IRW ALEVB DA
```

```
111655              T:=1; GO GORE1; *)FILL
111663           FI; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT
111671           A\/CCHAR; *IRW ALEVB DA
111673           IF D-1=0 THEN T:=2; GO GORE1 FI; IF X+1=L THEN X:=0 FI; *LBYT
111705           A SHZ 10
111706           IF D-1=0 THEN
111711                 *IRW ALEVB DD
111712                 T:=3; GO GORE1
111714           FI; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT
111722           A\/CCHAR; *IRW ALEVB DD
111724           IF D-1=0 THEN T:=4; GO GORE1 FI; IF X+1=L THEN X:=0 FI
111735           A:=D=:BHOLD; CFREE+4=:CFREE; *ION; IOF
111744           BHOLD=:D; *LBYT
111747           A SHZ 10
111750           IF D-1=0 THEN
111753                 *IRW ALEVB DL
111754                 T:=1; GO GORE2
111756           FI; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT
111764           A\/CCHAR; *IRW ALEVB DL
111766           IF D-1=0 THEN T:=2; GO GORE2 FI; IF X+1=L THEN X:=0 FI; *LBYT
112000           A SHZ 10
112001           IF D-1=0 THEN
112004                 *IRW ALEVB DX
112005                 T:=3; GO GORE2
112007           FI; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT
112015           A\/CCHAR; *IRW ALEVB DX
112017           D-1; T:=4
112021   GORE2:  A:=T+4
112023   GORET:  *IRW ALEVB DT
112024           A:=D=:BHOLD; CFREE+T=:CFREE
112031           IF X+1=L THEN X:=0 FI; X=:HENTE
112036           GO M8RET
112037
112037   GORE1:  A:=T; GO GORET
112041   RBUS
112042
112042
112042   %-=============================================================================
112042   % 39.11       B 4 I N T E R M   -   I B 4 I N T E R M
112042   %
112042
112042   SUBR B4INTERM,IB4INTERM
112042   INTEGER CCHAR
112043   IB4INTERM: T:=BUFST; GO FELLS
112045   B4INTERM: CALL SETBFPAGE; *PON
112047           IF FLAGB BIT 5LSTA THEN TERO2=:DERROR; GO TERWDX FI
112055           IF BHOLD<10 GO TERWDX              % NOT ENOUGH DATA IN BUFFER
112061           A:=B/\176000+BUFST=:T
112065   FELLS:  IF HENTE NBIT "0" THEN GO EVEN FI
112071           X:=HENTE; MAX=:L; *LBYT                                  % ( 1.BYTE)
112075           A SHZ 10; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT     % ( 2.BYTE)
112104           A\/CCHAR; *IRW ALEVB DA
112106           IF X+1=L THEN X:=0 FI; *LBYT                             % ( 3.BYTE)
112113           A SHZ 10; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT     % ( 4.BYTE)
112122           A\/CCHAR; *IRW ALEVB DD
112124           IF X+1=L THEN X:=0 FI; *LBYT                             % ( 5.BYTE)
112131           A SHZ 10; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT     % ( 6.BYTE)
112140           A\/CCHAR; *IRW ALEVB DL
112142           IF X+1=L THEN X:=0 FI; *LBYT                             % ( 7.BYTE)
112147           A SHZ 10; A=:CCHAR; IF X+1=L THEN X:=0 FI; *LBYT     % (10.BYTE)
```

```
112156          A\/CCHAR; *IRW ALEVB DX
112160          T:=10; A:=T; *IRW ALEVB DT
112163          BHOLD-T=:BHOLD; CFREE+T=:CFREE
112171          IF X+1=L THEN X:=0 FI; X=:HENTE; *ION
112177          GO M8RET
112200  *)FILL
112205
112205  EVEN:   A SHZ -1+T=:X; MAX SHZ -1+T=:L
112214          X.S0; *IRW ALEVB DA
112216          IF X+1=L THEN X:=T FI; X.S0; *IRW ALEVB DD
112224          IF X+1=L THEN X:=T FI; X.S0; *IRW ALEVB DL
112232          IF X+1=L THEN X:=T FI; X.S0; *IRW ALEVB DX
112240          BHOLD-10=:BHOLD; CFREE+10=:CFREE; A:=10; *IRW ALEVB DT
112250          IF X+1=L THEN X:=T FI; A:=X-T SHZ 1=:HENTE
112260          *ION
112261          GO M8RET
112262  RBUS
112263
112263  *"8STRN+8N500
"112263   %=============================================================================
112263  %       P T 3 0 S T E R M
112263  %
112263  % OUTSRTING TO TERMINALS
112263  %
112263  SUBR PT30STERM
112263
112263  DISP 0; DOUBLE CDBUADR=CBUADR,DD1=D1; PSID
112263  DISP 5REG; TRIPLE ZF1RG,ZF4RG,ZF7RG; PSID
112263  DISP -1; INTEGER M1; PSID
112263  INTEGER CDF,CDFADDRF
112265  INTEGER POINTER IPIT3:=177600+5BFPAGE+5BFPAGE
112266  INTEGER POINTER IPIT0:=177000+5BFPAGE+5BFPAGE
112267  DOUBLE POINTER DPIT0=IPIT0,DPIT3=IPIT3
112267
112267  P30STERM: X=:CDF
112270          T:=X.TDFPHPAGE=:D; A:=162000; AD=:DPIT3=:DPIT0
112275          T SH 10; RTREF.WINDOW/\377\/T=:X.WINDOW
112303          CDF.TDFLGADDR/\1777+"5BFPAGE*2000"=:X=:CDFADDR
112311          CALL COPTDFIELD
112312          A/\174770\/3002=:RTREF.ACTPRI/\3773; *TRR PCR; BSET ZRO; PION
112322          IF B="DEMFIELD" THEN X:=RTREF; CALL BRELEASE FI
112327          CDFADDR=:B; MLEV; *MST PIE
112333          CALL PONIOF
112334          NOCHAR SHZ -1; A-=:CNOCHAR
112340          MINBHOLD BONE 5BLOC=:MINBHOLD
112343
112343  LOOP:   T:=B SHZ -12 SH 12; FYLLE SHZ -1+BUFST+T=:D
112353          MAX SHZ -1+BUFST; T+A
112357          X:=CBUADR; BHOLD=:SBHOLD; 0=:BHOLD; CFREE SHZ -1=:L; *ION
112367          CNOCHAR-=:XNOCHAR; IF A=0 GO MBFERDIG
112373          IF FYLLE BIT 0 GO FAR NOTEVEN
112376          FOR CNOCHAR DO
112376             IF L=0 GO FAR NOSPAC; *BSET ONE
112401             X.S0; *BSET ZRO
112403             X+1:=:D; A=:X.S0
112406             L-1; IF X+1=T THEN A:=B/\176000+BUFST=:X FI; X:=:D
112417          OD
112421  FERDIG:
112421          CNOCHAR+XNOCHAR SH 1+SBHOLD=:SBHOLD
112426          IF NOCHAR BIT 0 THEN
```

```
112431              IF L=0 AND CFREE NBIT 0 THEN
112436                 A:=D/\1777-BUFST SH 1=:FYLLE
112443  INFERDIG:      CALL PONIOF; SBHOLD=:BHOLD; GO GGSTUPR; *)FILL
112466              FI; *BSET ONE
112467              T:=X.S0 SHZ -10 SH 10; *BSET ZRO
112473              D.S0/\377\/T=:X.S0; A:=D/\1777-BUFST SH 1
112504              IF A+1=MAX THEN A:=0 FI; MIN SBHOLD
112512           ELSE
112513              A:=D/\1777-BUFST SH 1
112517           FI; A=:FYLLE
112520
112520  GORET:   CALL PONIOF
112521           SBHOLD=:BHOLD; MAX-BHOLD=:CFREE
112526           MINBHOLD BZERO 5BLOC=:MINBHOLD; CALL STDEV
112532           A:=B+"ZOPRG"=:D; T:="30WFIELD"; 11=:L; *MOVNN
112541           "30WFIELD-5REG"=:B
112543           0=:ZAREG; GO RET
112545
112545  MBFERDIG: IF FYLLE BIT 0 THEN
112550               A=:D:=B/\176000+BUFST=:T; GO XFERDIG
112556            FI; GO FERDIG
112557
112557  *)FILL
112567
112567  INTEGER AREG,DREG; DOUBLE DADRG=AREG
112571  INTEGER 3PIT:=177600+5BFPAGE+5BFPAGE
112572  INTEGER POINTER P3PIT=3PIT
112572
112572  PONIOF:  *PIOF
112573           IF P3PIT=0 THEN
112575                X=:DREG; RTREF.WINDOW SHZ -10=:3PIT.S1; 162000=:X.S0; X:=DREG
112606           FI; *PON
112607           EXIT
112610
112610  NOSPACE: A:=D/\1777-BUFST SH 1=:FYLLE
112615  GOSTUPR: CALL PONIOF; CNOCHAR+XNOCHAR SH 1+SBHOLD=:BHOLD
112623  GGSTUPR: MAX-BHOLD=:CFREE; X=:CBUADR
112627           A:=B; AD=:DADRG
112631           TDRADDR=:B; X:=RTRES; CALL WDATA; AREG=:B; CALL STDEV
112640           "STUPR"; *IRW MLEVB DP
112642           AD:=DADRG; MLEV; *MST PIE; MST PID; ION
112647           CALL PONIOF
112650           GO FAR LOOP
112651
112651  NOTEVEN: FYLLE=:D
112653        A:=B/\176000+BUFST=:T
112657        FOR CNOCHAR DO
112657           IF L=0 THEN A:=D=:FYLLE; GO GOSTUPR FI; *BSET ONE
112665           X.S0 SHR 10; X:=:D; *BSET ZRO
112671           *SBYT
112672           IF MAX=X+1 THEN X:=0 FI; *BSET ONE
112700           X:=:D; X.S0; X:=:D; *BSET ZRO
112704           *SBYT
112705           IF MAX=X+1 THEN X:=0 FI; L-1; X:=:D; X+1
112715        OD
112717  XFERDIG: CNOCHAR+XNOCHAR SH 1+SBHOLD=:SBHOLD
112724        IF NOCHAR BIT 0 THEN
112727           IF L=0 AND CFREE NBIT 0 THEN A:=D=:FYLLE; GO FAR INFERDIG FI
112737           *BSET ONE
112740           X.S0 SHZ -10; X:=D; *BSET ZRO; SBYT
```

```
112745              IF X+1=MAX THEN X:=0 FI
112752              MIN SBHOLD
112753           ELSE
112754              X:=D
112755           FI; X=:FYLLE
112756           GO FAR GORET
112757
112757    BBUS
112771    *'
"112771    $ELIB
112771    %=============================================================================
112771    % 40.8          X M S G / P I T 3      E N T R Y      P O I N T S
112771    %
112771    %=============================================================================
112771    %
112771    SUBR CLXMS,PFXMS,MFXMS,DRXMS,ZXRES,ZXRRS,ZXTRS,P2XMS,ZXCOM,XMSGA,XPFAD,XDHOM
112771
112771    % GENERAL PHILOSOPHY: LEVELS 5,10,12,13,14 (DEC) WILL ALWAYS HAVE PIT3 SET
112771    % AS NORMAL AND ALTERNATIVE
112771    % XMSG ALWAYS RUNS ON PIT3
112771    INTEGER POINTER HOME=?
112771
112771    XPFAD: INTEGER ZXH:=0
112772    XMSGA: INTEGER POINTER ZGO:=ZGO+1          % GENERALISED ENTRY POINT TO XMSG PIT3
112773                                               % X=CODE: 1=DRIVER, 2=POWERFAIL, 3=CLOCK
112773                                               %         4=MEMORYFAIL, 5=MONCALL
112773                                               %         6=HDLC INIT, 7= HDLC RCV
112773                                               %         8=HDLC XMIT
112773                                               % OLD X-REG IS SAVED IN D-REG
112773    DUM: -45=:T                                % DUM IF XMSG NOT ACTIVE
112775         IF X-1=0 THEN MIN "HOME"; FI          % DO EXITA IF DRXMSG
113001         EXIT
113002    XDCOM: A:=D; X=:D:=1;                      % DRIVER CALL. RECOVER A-REG. SAVE X IN D
113005    ZXCOM:
113005    COM:    *PON
113006            GO ZGO                           % ORIGINAL X IS IN A. CODE IS IN X
113007
113007    % CLOCK HANDLING (MONITOR LEVEL)
113007    INTEGER POINTER CLSAV,PFSAV,MFSAV,HISAV,HRSAV,HTSAV
113015    CLXMS: X=:D:=L=:"CLSAV":=A; 3632; *TRR PCR       % SWITCH TO PIT3
113023           X=:A:=3; CALL COM; 32; *POF; TRR PCR      % AND BACK TO PIT0
113031           GO CLSAV
113032    PFXMS: X=:D:=L=:"PFSAV":=2; CALL COM; *POF
113040           GO PFSAV
113041    MFXMS: X=:D:=L=:"MFSAV":=4; CALL COM; *POF
113047           GO MFSAV
113050    ZXRES: X=:D:=L=:"HISAV":=6; CALL COM; *POF
113056           GO HISAV
113057    ZXRRS: X=:D:=L=:"HRSAV":=7; CALL COM; *POF
113065           GO HRSAV
113066    ZXTRS: X=:D:=L=:"HTSAV":=10; CALL COM; *POF
113074           GO HTSAV
113075    P2XMS: IF "ZGO"="DUM" OR ZXH=0 THEN EXIT; FI; *IRW DP 50; EXIT AD1
113106
113106
113106    % ENTRY POINT (PIOF) FROM DIRECT DRIVER ROUTINE CALLS TO XMSG (TO XDFUE)
113106    XDHOM: INTEGER POINTER HOME;INTEGER LEV,SAVA,SAVX
113112    DRXMS: A=:D:=50:=:L=:"HOME"; *TRA STS  % GET INTERRUPT LEVEL. SAVE A-REG IN D!!
113117           IF A SHZ -5/\170<L              % LOWER THAN LEVEL 5?
113121           THEN A=:LEV\/3602; *TRR PCR     % SWITCH TO PIT3 (APT AND NOT)
```

```
113126                CALL XDCOM; *POF            % CALL XMSG ROUTINE ON PIT3
113130                A=:SAVA:=LEV; X=:SAVX:=12   % SAVE RETURN VALUES
113134                IF A\/2=X THEN CURPROG.ACTPRI/\3773; FI; *TRR PCR;
113143                SAVA; X:=SAVX;
113145           ELSE CALL XDCOM; *POF            % GO TO SPESIAL VERSION OF COM
113151           FI; GO HOME
113151    RBUS
113160
113160    SUBR XMSGY                        % XMSG MONITOR CALL HANDLER (EVEN WHEN NO XMSG)
113160    INTEGER XPVL,XPBIT:=100000
113162    INTEGER XIRRP(0); * IRR DP
113163    INTEGER XIRWP(0); * IRW DP
113164    INTEGER XIRWT(0); * IRW DT
113165
113165    % COME HERE FROM POFMONC VIA GOTAB
113165    XMSGY: T:=A; *TRA PVL                 % SAVE PAGING STATUS IN T, GET LEVEL
113167           A=:D/\170=:XPVL                % SAVE LEVEL
113172           IF T NBIT 16                   % WAS CALLING LEVEL IN POF?
113172           THEN A SHZ -3=:X; *EXR SD      % GET HIS P-REG
113177                CALL XFIPV; T:=0; XPVL    % SAVE IN SWITCH TABLE, T=PAGING STATUS
113202           ELSE T:=XPBIT                  % T=PON BIT, A=PREVIOUS LEVEL*10
113204           FI;
113204           X:=5; D:=T; CALL ZXCOM         % GOTO XMSG (A=PREVLEV*10,T=D=PAGING STAT)
113207           % RETURN HERE IF XMSG NOT RUNNING (D=0 IF PAGING WAS OFF)
113207           IF XPVL>40                     % DRIVER CALL?
113211           THEN  A SHZ -3=:X              % GET PAGING STATUS
113215                 IF 0=D                   % WAS PAGING OFF?
113215                 THEN X:=PVLAD(X); X.S2+1=:X.S2 % SET SWITCH TABLE AGAIN TO A SKIP
113223                 ELSE
113224                    XPVL\/XIRRP; *EXR SA  % GET HIS P-REG (AGAIN!)
113227                    A+1=:D;=XPVL\/XIRWP:=:D; *EXR SD % GIVE HIM A SKIP RETURN,WRITE HIS P-REG DIRECT
113235                 FI
113235           FI
113235           XPVL\/XIRWT=:D:=-45; *EXR SD   % SET HIS T-REG TO 'XMSG NOT RUNNING'
113242           GO RET14
113243    RBUS
113250
113260    *'8XMSG+8CXHD+8C1X2+8C2X2+8C3X2+8C4X2
```

```
"113250
113250    %=============================================================================
113250    %
113250    % 40.9       GENERAL BUFFER ALLOCATION PACKAGE - ZB... for XMSG and HDLC
113250    %
113250    %=============================================================================
113250    SUBR ZBINI,ZBGET,ZBREL
113250    %
113250    %       THIS PACKAGE IS BASED ON THE CERN SMO MEMORY ALLOCATION PACKAGE. IT
113250    % ALLOCATES BUFFERS OF SIZE MULTIPLE OF FOUR WORDS. THE FIRST WORD OF EACH
113250    % BUFFER IS USED BY THE ALLOCATION PACKAGE AND CONTAINS:
113250    %       BIT 0 : SET IF THE BUFFER IS FREE (ZBCFR)
113250    %           1 : SET IF PRECEDING BUFFER IS FREE (ZBPFR)
113250    %        2-15 : BUFFER LENGTH IN MULTIPLE OF FOUR WORDS
113250    %
113250    %       IF THE BUFFER IS FREE (BIT 0 SET), THE LAST WORD OF THE BUFFER CONTAINS
113250    % THE ADDRESS OF ITS START (FOR CONCATENATION OF FREED BUFFERS.)
113250    %
113250    %       THE USER OF THIS PACKAGE MUST PROVIDE OF BUFFER AREA DESCRIPTOR OF 4
113250    % WORDS, WHICH MUST BE POINTED AT BY X ON ENTRY TO THESE ROUTINES:
113250    %
113250    DISP 0
113250            INTEGER ZBBNK           % BANK NO WHERE BUFFER AREA LIES (=0 IF N10)
113250            INTEGER ZBSTR           % ADDRESS OF FIRST WORD USED IN BANK FOR BUFFERS
113250            INTEGER ZBEND           % ADDRESS OF LAST WORD USED IN BANK FOR BUFFERS
113250            INTEGER ZBSAV           % SAVE LOCATION FOR USER B-REG
113250    PSID
113250    SYMBOL ZBCFR=0,ZBPFR=1          % BITS IN THE HEADER (DO NOT CHANGE - SEE CODE)
113250    %
113250    %       FUNCTIONS PROVIDED MUST BE CALLED WITH X POINTING TO THE DESCRIPTOR.
113250    % THE RETURN WILL BE BY SKIP RETURN UNLESS A FATAL ERROR IS DETECTED IN WHICH
113250    % CASE A WILL CONTAIN AN ERROR CODE:
113250    SYMBOL ZBX01=1                  % BANK NO >< 0 AND CPSTA SAYS THIS IS A N10
113250    SYMBOL ZBX02=2                  % NOT ENOUGH BUFFER SPACE TO INITIALISE
113250    SYMBOL ZBX03=3                  % INCONSISTENCY IN BUFFERS FOUND BY ZBGET
113250    SYMBOL ZBX04=4                  % ILLEGAL USER BUFFER ADDRESS TO ZBREL
113250    %
113250    %       FUNCTIONS ARE:              (B -> BUFFER AREA DESCRIPTOR)
113250    %
113250    % ZBINI:       INITIALISE BUFFER AREA
113250    %              T=BANK NO, A=ADDR OF FIRST WORD TO USE, D=ADDR OF LAST WORD
113250    %
113250    % ZBGET:       GET BUFFER SPACE
113250    %              A=NO OF BYTES
113250    %       SKIP:  A:=USER BUFFER ADDRESS ( OR 0 IF NONE AVAILABLE )
113250    %              NB: This is the address of the first USABLE word in the buffer
113250    %
113250    % ZBREL:       RELEASE BUFFER SPACE
113250    %              A=USER BUFFER ADDRESS
113250    %
113250    INTEGER ZBMSK:=177774
113251    INTEGER POINTER ZBCPS:=CPSTA
113252
113252    ZBINI: X:=:B=:ZBSAV; T=:ZBBNK;                      % INITIALISE DESCRIPTOR
113255           IF T><0 AND X:=ZBCPS NBIT 5N100              % BANK><0 AND NOT N100
113260           THEN ZBX01; GO ZBOUT; FI                     % NOT ALLOWED
113264           X:=ZBMSK; A+3/\X=:ZBSTR; X/\D=:ZBEND         % ROUND START UP, END DOWN
113272           IF A+10>>=X THEN ZBX02; GO ZBOUT; FI         % NOT ENOUGH SPACE
113277           X-1; *STATX 0; SAA 2; STATX 10               % BACKPTR. LENG=0, PREV. FREE
113303           X:=ZBSTR; ZBEND-X BONE ZBCFR; *STATX 0       % SET LENGTH & FREE
```

```
113310          GO ZBOKR                                        % SKIP RETURN
113311
113311    ZBGET: X:=:B=:ZBSAV                                   % SAVE USER B-REG. B->DESCR.
113313          A SHZ -1; *BSKP ZRO SSM; RINC DA                % A=NUMBER OF USER WORDS
113316          A+4/\ZBMSK=:D; X:=ZBSTR                         % D=NUMBER OF WORDS NEEDED
113322          DO WHILE X><ZBEND                               % SCAN DOWN BUFFERS
113325                ZBX03; IF X>>T OR X<<ZBSTR GO ZBOUT       % BUFFERS DESTROYED!
113333                T:=ZBBNK; *LDATX 0                        % GET HEADER WORD
113335                IF A BIT ZBCFR AND A>>=D THEN             % SPACE AVAILABLE?
113341                      A BZERO ZBCFR :=:D; * STATX 0       % SET NEW LENGTH IN HEADER
113344                      % NOW HAVE: X=BUFAD, A=NEWSIZE, D=OLDSIZE, T=BANK NO
113344                      D-A; A+X:=:X                        % D=REMNANT, A=USER BUF, X=NEW
113347                      IF D=0                              % FILLED WHOLE BUFFER?
113347                      THEN D:=A; * LDATX 0                % GET HEADER OF NEXT BUFFER
113353                            A BZERO ZBPFR; * STATX 0      % CLEAR PREVIOUS FREE
113355                            A:=D                          % RECOVER USER ADDRESS
113356                      ELSE; * STATX 10                    % SAVE USER ADDR IN NEW BUF(1)
113360                            A:=D BONE ZBCFR; *STATX 0     % BUILD NEW HEADER
113363                            X=:A+D-1; * STATX 0           % SET BACK POINTER
113367                            X;=A; * LDATX 10              % RECOVER USER ADDRESS
113371                      FI;
113371                      A+1; GO ZBOKR                       % RETURN WITH USER ADDR.
113373                FI; A/\ZBMSK+X=:X                         % TRY NEXT
113376          OD;
113377          A:=0; GO ZBOKR                                  % NO SPACE LEFT
113401
113401    ZBREL: X:=:B=:ZBSAV; A-1=:X/\ZBMSK                    % CONVERT USER->BUFFER ADDR
113406          IF A><X OR A<<ZBSTR OR A>>=ZBEND GO ZBIOW       % CONSISTENCY CHECKING
113416          T:=ZBBNK; * LDATX 0                             % GET BUFFER HEADER
113420          IF A BIT ZBCFR THEN GO ZBIOW; FI                % SHOULD BE ALLOCATED
113423          A/\ZBMSK; X=:D+A; * LDATX 0                     % GET HEADER OF NEXT BUFFER
113427          IF A BIT ZBCFR THEN A/\ZBMSK; X+A; FI           % IF FREE FIND ONE AFTER
113433          X:=:D; * LDATX 0                                % GO BACK TO ORIGINAL.
113435          IF A BIT ZBPFR                                  % IF PREVIOUS FREE
113435          THEN X-1; * LDATX 0; COPY DX SA                 % GET POINTER TO START OF PREV
113442          FI
113442          % WE NOW HAVE X=START OF NEW (CONCATENATED) BUFFER, D=START OF NEXT
113442          A:=D-X BONE ZBCFR; * STATX 0                    % SET HEADER OF NEW
113446          X=:A:=D-1; * STATX 0; LDATX 10                  % SET BACK PTR, GET NEXT HEAD
113453          A BONE ZBPFR; * STATX 10                        % SET PREVIOUS FREE
113455    ZBOKR: L+1; GO ZBOUT                                  % SKIP RETURN SEQUENCE
113457    ZBIOW: ZBX04;                                         % ILLEGAL BUFFER ADDRESS
113460    ZBOUT: X:=ZBSAV:=:B; EXIT                             % RESTORE USER B AND X-REGS
113463    RBUS
113463    %==================== END OF BUFFER ALLOCATION ROUTINES (ZB...) ==============
113463
```

```
113463
113463    + 8LPPU
"113463    %=============================================================================
113463    % 33.10       L P P U T
113463    %
113463    @MAC

)9SCLC
113463   % LINE PRINTER COVERTING ROUTINES
113463   PTABL=*
113463   PTABL<PTABL 27
113463   )ZERO
113463   % INSERT INPUT AND OUTPUT VALUES HERE
113463   %
113463   160;30
113465   44;135
113467   135;134
113471   43;133
113473   100;134
113475   0
113476   PTABL 20/
113503   % IOTRANS ROUTINE FOR LINE PRINTER
113503   % THE CHARACTERS IN PTABL ARE CONVERTED
113503   LPPUT, COPY  SX DD
113504          LDX   (PTABL
113505          BSET ZRO 70 DA
113506          LDT ,X
113507          SKP   DT UEQ 0
113510          JMP   PCON1
113511          SKP   DT UEQ SA
113512          JMP   * 3
113513          AAX   2
113514          JMP   LPPUT 3
113515          LDA   1 ,X
113516   PCON1, SAT   0
113517          SAX   -10
113520          BSET  ZRO 70 DA
113521          SHA   10
113522          JAP   *+2
113523          AAT   1
113524          SHA   ROT 1
113525          JNC   *-3
113526          BSKP  ZRO 00 DT
113527          BSET  ONE 70 DA
113530          COPY  SD DX
113531          JMP I (TRTPU
113532   )FILL
113534   )9RCLC
)9SLPL113534    *"
"113534
113534    %=============================================================================
113534    % 37.15       D P R I N T
113534
113534    % SUBROUTINE TO PUT TEXT DIRECTLY INTO TERM.1-BUFFER
113534    % ENTRY:      A=TEXT POINTER
113534    SUBR DPRINT
113534    INTEGER TRTX,LREG,BREG,XREG
113540    DPRINT:  A=:TRTX:=L=:LREG:=B=:BREG:="DT01W"=:B; X=:XREG:=0; *IOF
113552    @LIB CXCPU
113552           CALL SPT3WINDOW
```

```
113553          DO T:=TRTX; *LBYT
113555          WHILE A><##'
113560                *PON; JPL I (CXRBP; POF; AAX 1
113564          OD; *PON
113566          -1=:TMR; CALL RPIT3              % OUTPUT DRIVER IS STARTED BY TIMER
113571  @ELIB
113571  @ELIB CXCPU-,
113571          *ION
113572          X:=LREG=:L:=BREG=:B:=XREG; EXIT
113600  RBUS
113604
113604  @ELIB CXCPU
113604  %=============================================================================
113604  %              R P I T 3   -   S P T 3 W I N O W
113604  %
113604  %  CLEAR AND SET WINDOW ON PIT3 FOR MONITOR LEVEL
113604  %
113604  SUBR RPIT3,SPT3WINDOW
113604
113604  INTEGER POINTER IPIT3:=177600+5BFPAGE+5BFPAGE
113605  DOUBLE POINTER DPIT3=IPIT3
113605
113605  RPIT3: *PIOF
113606          0=:IPIT3; A:=32; *TRR PCR
113611          EXIT
113612  SPT3WINDOW: A:=TDFPHPAGE=:D:=162000; AD=:DPIT3
113616          TDFLGADDR/\1777+"5BFPAGE*2000"=:B
113622          3632; *TRR PCR
113624          EXIT
113625  RBUS
113631  @ELIB
113631
113631
113631  *9EPT3=*                        % END OF CODE MAPPED INTO PAGE TABLE 3
113631
113631
113631  %=============================================================================
113631  % 37.1          C T R D I S K   B U S Y E   M D R I V   P F E I L
113631  %              C O O P T   D R F E I L
113631  %
113631  % LEVEL 11 ROUTINE TO PERFORM DRUM/DISK TRANSFERS
113631  % ACTIVATED BY MTRANS, WITH B=DATAFIELD, X=ABSTR PAR.LIST
113631
113631
113631  SUBR CTRDISK,BUSYE,MDRIV,PFEIL,COOPT,DRFEIL
113631
113631  DISP 0; DOUBLE POINTER DP1=P1; PSID
113631  DISP 12; INTEGER TMRFLG; PSID
113631  DISP 0; DOUBLE DITARG=CTRG,DIDXRG=CDRG; PSID
113631  INTEGER  CBLDA(0); *BLDA 00 DT
113632  INTEGER CUN=?
113632
113632  CTRDISK: X:=:B; CALL PICKFPAR; IF A=0 AND D=0 THEN CALL ERRFATAL FI
113640          AD=:X.MEMAD; A/\3; A SH 14+T=:X.CTRG
113645          A SHZ -6/\7 SH 3\/CBLDA
113651          T:=X.M2UNTYP; *EXR SA
113653          IF K THEN
113655             X.CTRG BONE 16=:X.CTRG; 0=:X.CARG        % IF PHOENIX DISC
113661          ELSE
113662             X.CTRG/\7000 SHZ -11=:X.CARG
```

```
113666          FI
113666          IF X.CTRG/\77>=60 AND A<=63 THEN
113676             X.CTRG/\177717=:X.CTRG; CALL PICKVLPAR; AD=:X.CADRG
113703          ELSE
113704             CALL PICKLPAR; A=:X.CDRG
113706          FI; T=:X.CXRG
113707          X=:B; CTRG=:TRGINI
113712          IF A/\77=36 THEN                                    % READ ELEMENT IN "DISC-LAYOUT-TABLE"
113716             0=:HSTAT; AD:=MEMAD; *EXAM                       % T=FORMAT NUMBER
113721             IF 40>T AND DISPE(T)><0 THEN                     % 37 IS MAX FORMAT NUMBER
113727                T:=MEMA1; X:=MEMA2; B=:L:=A; "DILEZ"=:D
113735                DO WHILE D><0                                 % COPY "DISC LAYOUT TABLE" ELEMENT TO "DMA-BUFF
113737                   S0; *STATX
113741                   X+1; D-1; B+1
113744                OD; B:=L
113746             FI; GO FAR FIN; *)FILL
113763          FI
113763          IF A=42 AND "TRNSF"><"BDISK" GO FAR FIN    % READ FORMAT NOT LEGAL IN DRIVER
113772                                                     % BUT GIVE NO ERROR MESSAGE
113772          IF TMR><0 THEN CALL ID11 FI; 0=:TMRFLG
113776    +"8DILG
"11377o           IF "DFDIL".DILGFLAG BIT DILCOUNT THEN
114002             IF A BIT DAC1CONTROLLER AND B><DLALOGDV GO L1       % DO'NT COUNT THIS CONTROLLER
114007             IF A BIT DAC1UNIT AND CTRG SHZ -6/\7><DLAUNIT GO L1 % DO'NT COUNT THIS UNIT
114017             *MIN 2XNDA,X; SKP; MIN 1XNDA,X; JMP * 1             % COUNT DISC ACCESSES
114023             IF CTRG/\77=1 THEN                                  % COUNT WRITE ACCESSES
114030                *MIN 2XNWD,X; SKP; MIN 1XNWD,X; JMP *+1
114034             FI
114034           FI; GO L1; *)FILL
114043    L1:    IF X.DILGFLAG BIT DILSTART AND A BIT DILBOK THEN    % DISC LOG STARTED?
114050             A=:D; X:="DIDLOG"
114052             DO WHILE X.S0><-1                                   % FIND LOGICAL UNIT
114056                IF A=B GO LGFOUND
114060                X+2
114061             OD; GO NOTLOG
114063    INTEGER CLUN
114064    LGFOUND:  X.S1=:CLUN; X:="DFDIL"                            % LOGICAL UNIT OF DISC
114067             IF A:=D/\37=0 GO DOLOG                             % LOG EVERY ACCESS
114072             IF D BIT DIL1CONTROLLER THEN                       % LOG ONLY ONE CONTROLLER?
114074                IF X.DLLOGDV><CLUN GO NOTLOG                    % YES, IS IT THIS CONTROLLER?
114100             FI
114100             IF D BIT DIL1UNIT THEN                             % LOG ONLY ONE UNIT?
114102                IF CTRG SHZ -6/\7><X.DLDRIVE GO NOTLOG          % YES, IS IT THIS UNIT?
114110             FI
114110             IF D BIT DILWACCESS THEN                           % LOG ONLY WRITE ACCESSES?
114112                IF CTRG/\77><1 GO NOTLOG
114117             FI
114117             IF D BIT DILRACCESS THEN                           % LOG ONLY READ ACCESSES?
114121                IF CTRG/\77><0 GO NOTLOG
114124             FI
114124             IF D BIT DILLIMIT THEN                             % LIMITED AREA TO LOG?
114126                AD:=CADRG; A=:L; D=:T; AD:=X.DILFADDR           % YES, TEST DISC ADDRESS
114132                IF L<<A GO NOTLOG; IF L=A AND T<<D GO NOTLOG
114140                AD:=X.DILGLADDR
114141                IF L>>A GO NOTLOG; IF L=A AND T>>D GO NOTLOG
114147             FI
114147    DOLOG:   T:="DFDIL".DILBANK; A:=X.DILGFLAG=:L:=X.DILBPNT    % SET UP DISC LOG RECORD
114154             IF L BIT DILSMALL THEN
114156                A+4=:X.DILBPNT; A-4
114161             ELSE
```

```
114162                A+10=:X.DILBPNT; A-10
114165             FI; X:=A
114166             AD:=DITARG; *STDTX
114170             IF L BIT DILSMALL THEN
114172                CLUN=:D; A:=CDRG; *STDTX 20
114176             ELSE
114177                AD:=DIDXRG; *STDTX 20
114201                AD:=MEMAD; *STDTX 40
114203                A:=RTRES=:D:=CLUN; *STDTX 60
114207             FI; CALL WRDILOG; GO NOTLOG; *)FILL                % WRITE DISC LOG BUFFER
114217          FI
114217    *"
"114217     NOTLOG: 0=:HSTAT; 1=:DIFTCOUNT                            % ONE RESTART BY TIMRT IS MAX
114222    *"8DILG
 114222           IF "TRNSF"="BDISK" THEN 0=:SPACO=:CORCU=:SRTRY=:SWTRY FI
114232    *"
"114232     LOOP:                                                     % FOR TRANSF AND COMPARE
114232                TACNS=:TACOU
114234                FOR TACOU DO                                    % FOR EACH ERROR
114234                   AD:=MEMAD=:CMADR
114236    CREST:       CTADRG=:TADRG; X:=CXRG=:XRG
114242                   DO                                           % FOR EACH PHYSICAL TRANSFER
114242                      TTMR=:TMR; A:=ARG
114245                      CALL TRNSF; GO ERROR; GO BUSY; GO FINISH
114251    BUSYE:            TAD=:TADRG; X=:XRG; CALL ID11             % WAIT FOR INTERRUPT
114254                      IF TMR=0 THEN                             % TIMEOUT
114256    *"8DILG                                                                    --
"114256                          D:=B; X:=DEDFADDR:=:B
114261                        IF X.CTRG SHZ -6/\7<4 THEN              % A=DISC UNIT NUMBER
114267                           IF T:=X.CTRG BIT "0" THEN            % WRITE ACCESS
114272                              MIN DIEWTMOUT(A); 0/\0            % INCREMENT WRITE-ACCESS TIMEOUT COUNTER
114275                           ELSE
114276                              MIN DIERTMOUT(A); 0/\0            % INCREMENT READ-ACCESS TIMEOUT COUNTER
114301                           FI
114301                        FI; B=:D
114302    *"
"114302                          IF HSTAT><0 GO FAR ERTMOUT
114305                        -1=:HSTAT
114307                      FI
114307
114307    MDRIV:            TADRG; X:=XRG
114311                   OD
114312
114312    PFEIL:           IF X NBIT 16 AND DIFTCOUNT><0 THEN         % NOT ON CYLINDER
114316                      A-1=:DIFTCOUNT; CALL ID11; GO CREST       % TIMRT REACTIVATE DRIVER
114322                   FI
114322    DRFEIL:          X BONE 4 =:HSTAT; AERRB\/X=:AERRB; MIN ERCNT; 0/\0
114331    *"8DILG
"114331                     IF CTRG SHZ -6/\7-4<0 THEN                % SAVE ERROR INFO FOR EACH UNIT
114336                      A+4=:X=:CUN:=DEDFADDR=:D:=HSTAT; B:=:D
114345                      A=:DIUEXRG(X); T=:DIUETRG(X)
114347                      T:=D.HSTAT
114351                      IF X.CTRG BIT "0" THEN                    % WRITE FUNCTION
114354                         MIN DIEWCOUNT(CUN); 0/\0
114357                         DIEWOR(X)\/T=:DIEWOR(X)
114362                      ELSE
114363                         MIN DIERCOUNT(CUN); 0/\0
114366                         DIEROR(X)\/T=:DIEROR(X)
114371                      FI; CALL UPDIERR; B:=D; X:=HSTAT
114374                   FI
```

```
114374   *"
"114374                    IF SERRB/\X><0 GO FIN  %SERIOUS
114377                    IF X BIT 12 THEN TRGINI=:CTRG FI
114403                 OD; GO FIN; *)FILL
114413   *"8DILG
"114413    INTEGER CUN,XSPACO,XCORCU,XSRTRY,XSWTRY
114420   *"
"114420
114420   COOPT:         X=:HSTAT
114421   *"8DILG
"114421                 IF CTRG SHZ -6/\7<4 THEN A=:X:=DEDFADDR:=B=:D; CALL UPDIERR; B:=D FI
114435   *"
"114435                 IF COMFL=0 GO FIN
114437                 IF TRG/\7=3 GO FIN                    % COMPARE HAS BEEN EXECUTED
114444                 IF "TRNSF"="BDISK" GO FIN             % NO COMPARE ON ECC DISKS
114450                 IF CTRG/\77>1 GO FIN                  % COMPARE FOR FUNCTION 0 AND 1
114455                 177770/\TRG\/3=:CTRG                  % SET COMPARE
114461         GO FAR LOOP; *)FILL
114470
114470   FIN:
114470         IF CTRG/\77=42 THEN                                    % READ FORMAT
114475            CTRG SHZ -6/\7=:X                                   % X=UNIT NUMBER
114501            X:=HTABL(X); T:=X.DISPN; AD:=MEMAD; *DEPO           % RETURN FORMAT NUMBER IN FIRST LOCATION OF "DM
114505            0=:HSTAT                                            % NO ERROR MESSAGE
114506         FI
114506
114506         0=:TMR; IF RTRES><0 THEN CALL RTACT FI
114512         IF HSTAT BIT 4 THEN
114515               HDEV; T:=TRG SH 7 SHZ -15; CALL 9ERR(#20)
114523               DRG; T:=HSTAT; CALL 9ERR(#21)
114527         FI
114527   FINE:  CALL ID11; A+1; GO ERR22
114532   *)FILL
114540
114540   ERTMOUT: HDEV; T:=TRG SH 7 SHZ -15; CALL 9ERR(#26)          % DEVICE TIMEOUT
114546            HSTAT BONE 4=:HSTAT; GO FINE
114552
114552   *"8DILG
"114552    UPDIERR: B:=:D
114553            IF "BDISK"="TRNSF" THEN
114557               A:=SPACO=:XSPACO:=CORCU=:XCORCU:=SRTRY=:XSRTRY:=SWTRY=:XSWTRY
114567               B:=:D
114570               DIERRTRY(X)+XSRTRY=:DIERRTRY(X)
114573               DIEWRTRY(X)+XSWTRY=:DIEWRTRY(X)
114576               DIECORCU(X)+XCORCU=:DIECORCU(X)
114601               DIESPACO(X)+XSPACO=:DIESPACO(X)
114604            ELSE
114605               B:=:D
114606            FI; EXIT
114607   *"
"114607    RBUS
```

```
114611
114611
114611    *8DILG
 114611    %=============================================================================
114611    %       DATAFIELD ADDRESSES AND LOGICAL UNITS OF ALL DISCS
114611    %
114611    @ICR;
114611    INTEGER ARRAY DIDLOG:=(
114611    BIGD1,1100,
114613    BIGD2,1207,
114615    BIGD3,565,
114617    BIGD4,566,
114621    WIGD1,1224,
114623    WIGD2,1231,
114625    DRFIE,502,
114627    DRFI2,1104,
114631    GIGD1,1100,
114633    GIGD2,1207);
114635    @CR;
114635
114635    %=============================================================================
114635    %  DISC ERROR DATAFIELD
114635    %
114635    *8DI1+8BD1+8GD1+8WD1+8MD1 8DILG
"114635    INTEGER ARRAY DEDF1(60)
114715    *8DI2+8BD2+8GD2+8WD2+8MD2 8DILG
"114715
114715    %=============================================================================
114715    %             W R D I L O G
114715    %
114715    % SUBROUTINE TO START THE RT-PROGRAM FOR WRITING TO THE DISC LOG FILE
114715    %
114715    % ALL REGISTERS EXCEPT THE B-REGISTER ARE USED IN THIS ROUTINE
114715    %
114715    SUBR WRDILOG
114715    WRDILOG:
114715            IF "DFDIL".DILBPNT/\1777=0 THEN                          % BUFFER #2 IS FULL
114721               X.DILGFLAG BONE 2DILBFULL=:X.DILGFLAG                 % MARK BUFFER #2 TO WRITE
114724               X.2DIBADDR=:X.DILBPNT                                 % SET NEW BUFFER ADDR
114726               "RTDIL".STATUS BZERO 5WAIT=:X.STATUS; 1=:MTOR         % START RT-PROGRAM
114734               EXIT
114735            FI
114735            IF A/\777=0 THEN                                         % BUFFER #1 IS FULL
114737               X.DILGFLAG BONE 1DILBFULL=:X.DILGFLAG                 % MARK BUFFER #1 TO WRITE
114742               X.2DIBADDR+1000=:X.DILBPNT                            % SET NEW BUFFER ADDR
114745               "RTDIL".STATUS BZERO 5WAIT=:X.STATUS; 1=:MTOR         % START RT-PROGRAM
114753            FI; EXIT
114754    RBUS
114762
114762    %=============================================================================
114762    %             D V D I L O G
114762    %
114762    % ROUTINE ON LEVEL 11 FOR DISC LOG
114762    %
114762    % ENTRY:      B=DISC LOG DATAFIELD
114762    %             X=ABSTR PARAMETER LIST
114762    %
114762    SUBR DVDILOG
114762
114762    INTEGER CCMBNK=?,CCMADDR=?
```

```
114762   DOUBLE POINTER DCCMADR:=CCMBNK
114763   INTEGER CFUNC
114764
114764   D/DILOG: X:=:B; CALL PICKFPAR
114766            AD=:DCCMADR; T=:CFUNC; X=:B
114771            IF CFUNC>>13 GO FAR ERET                    % ILLEGAL FUNCTION
114775   @ICR;
114775            A GOSW FAR ERET,FUN1,FUN2,FAR FUN3,FUN4,FUN5,
115004                   FUN6,FUN7,FAR FUN10,FAR FUN11,FAR FUN12,FAR FUN13
115011   @CR;
115012
115012   INTEGER POINTER PNCMBNK:=CCMBNK,PNCMADR:=CCMADR
115014
115014   FUN1:                            % WRITE DISC LOG RECORD
115014   FUN2:                            % WRITE DISC LOG RECORD, CLEAR REST OF DISC LOG FILE BUFFER
115014                                    % AND WRITE DISC LOG BUFFER TO LOG FILE
115014           IF A:=DILGFLAG=:L NBIT DILSTART OR A NBIT DILBOK GO FAR ERET
115022           T:=PNCMBNK; X:=PNCMADDR; IF T=0 AND X=0 GO FAR ERET; *LDDTX
115031           T:=DILBANK; X:=DILBPNT; *STDTX
115034           T:=PNCMBNK; X:=PNCMADDR; *LDDTX 20
115037           T:=DILBANK; X:=DILBPNT; *STDTX 20
115042           IF L NBIT DILSMALL THEN
115044              T:=PNCMBNK; X:=PNCMADDR; *LDDTX 40
115047              T:=DILBANK; X:=DILBPNT; *STDTX 40
115052              T:=PNCMBNK; X:=PNCMADDR; *LDDTX 60
115055              T:=DILBANK; X:=DILBPNT; *STDTX 60
115060              X+4=:DILBPNT
115062           FI
115062           X+4=:DILBPNT
115064           IF CFUNC=2 THEN
115070              D:=0; T:=DILBANK
115072              DO WHILE A:=X/\777><0
115075                 A:=0; *STDTX; AAX 2
115100              OD; X=:DILBPNT
115102           FI; CALL WRDILOG
115103           GO FAR OKRET
115104   *)FILL
115116
115116   FUN4:    IF DILGFLAG NBIT DILDEFINED GO FAR ERET    % START DISC LOG
115121           A BONE DILSTART=:DILGFLAG
115123   L1:     X:="XDILD-MLINK":=:B; CALL XRTACT; X=:B
115127           GO FAR OKRET
115130
115130   FUN5:    DILGFLAG BZERO DILSTART=:DILGFLAG          % STOP DISC LOG
115133           GO L1
115134
115134   INTEGER CDILGFLAG
115135   FUN6:    0=:CDILGFLAG                               % START DISC-ACCESS-COUNTER
115136           T:=CCMBNK; X:=CCMADDR; IF T=0 AND X=0 GO FAR ERET; *LDATX  % FLAG
115145           A=:CDILGFLAG
115146           IF A BIT "1" THEN                          % COUNT FOR ONE UNIT ONLY
115150              *LDATX 20                               % UNIT NUMBER
115151              IF A>>3 GO FAR ERET
115154              A=:DLAUNIT
115155              CDILGFLAG BONE DAC1UNIT=:CDILGFLAG
115160           FI
115160           IF CDILGFLAG BIT "0" THEN                  % COUNT FOR ONE CONTROLLER ONLY
115163              *LDATX 10
115164              CALL LOGPH; IF A=0 GO FAR ERET
115167              CALL CCHDILOG; GO FAR ERET
```

```
115171              A=:DLALOGDV
115172              CDILGFLAG BONE DAC1CONTROLLER=:CDILGFLAG
115175           FI; DILGFLAG BZERO DAC1UNIT BZERO DAC1CONTROLLER
115200           A BONE DILCOUNT\/CDILGFLAG=:DILGFLAG
115203           GO FAR OKRET
115204
115204    FUN7:  DILGFLAG BZERO DILCOUNT=:DILGFLAG          % STOP DISC ACCESS COUNTER
115207           GO FAR OKRET
115210
115210    FUN10: A:=0=:D; AD=:DXNDACCESSES=:DXNWDACCESSES  % CLEAR DISC ACCESS COUNTER
115214           GO FAR OKRET
115215
115215    FUN11: T:=CCMBNK; X:=CCMADDR                     % READ DISC ACCESS COUNTER
115217           AD:=DXNDACCESSES; *STDTX
115221           AD:=DXNWDACCESSES; *STDTX 20
115223           GO FAR OKRET
115224    *)FILL
115232
115232    INTEGER CCMBNK,CCMADDR
115234
115234    CCHDILOG: A=:D; X:="DIDLOG"
115236           DO WHILE X.S0><-1
115242              IF A=D THEN EXITA FI
115245              X+2
115246           OD; EXIT
115250
115250    FUN12:                                           % READ LAST DISC-ERROR INFO
115250           T:=CCMBNK; X:=CCMADDR; *LDATX             % A=:LOGICAL UNIT
115253           CALL LOGPH; IF A=0 GO FAR ERET
115256           CALL CCHDILOG; GO FAR ERET
115260           IF A."TRNSF"><"BDISK" GO FAR ERET
115265           A.S2=:L; T:=CCMBNK; X:=CCMADDR
115272           X:=:L; X.DS0; X+2:=:L; *STDTX
115277           X:=:L; X.DS0; X+2:=:L; *STDTX 20
115304           X:=:L; X.DS0; X+2:=:L; *STDTX 40
115311           X:=:L; X.DS0; X+2:=:L; *STDTX 60
115316           X+10:=:L; X.DS0; X+2:=:L; *STDTX
115324           X:=:L; X.S0; X:=:L; *STATX 20
115330           GO FAR OKRET
115331    *)FILL
115337
115337    INTEGER CUN,NNC=CUN
115340    FUN13:                                           % READ DISC ERROR STATUS
115340           T:=CCMBNK; X:=CCMADDR; *LDDTX             % A= LOGICAL DEV. NO; D=UNIT NO.
115343           IF A=0 OR D>>3 GO FAR ERET
115347           A:=:D=:CUN:=D; CALL LOGPH; IF A=0 GO FAR ERET
115355           CALL CCHDILOG; GO FAR ERET
115357           A:=A.DEDFADDR+CUN=:D; D:=:B
115364           T:=CCMBNK; X:=CCMADDR
115366           -14=:NNC
115370           FOR NNC DO
115370              S0; *STATX
115372              B+4; X+1
115374           OD; B:=D; GO FAR OKRET
115400
115400    INTEGER POINTER PCCMBNK:=CCMBNK
115401
115401    % DEFINE DISC LOG
115401    %
115401    FUN3:  IF DILGFLAG BIT DILSTART GO ERET
```

```
115404          A BZERO DILDEFINED=:DILGFLAG
115406          T:=PCCMBNK; X:=CCMADDR; *LDDTX              % FIRST DISC ADDR OF LOG FILE
115411          AD=:DILDADDR; *LDATX 20                     % A=NO. OF DISC SECTORS PER TRACK
115413          IF A><2 GO ERET; A=:DILNSEC
115417          T:=PCCMBNK; *LDDTX 30                       % NUMBER OF PAGES IN DISC LOG FILE
115421          AD=:DILLADDR                                % LAST LEGAL DISC ADDR IN LOG FILE
115422          IF A<<DIL1DADDR GO ERET; IF A=T AND D<<DIL2DADDR GO ERET
115432          T:=PCCMBNK; *LDDTX 50                       % A=LOG.DEV, D=DRIVE NUMBER
115434          AD=:DDILFLOG; CALL LOGPH; IF A=0 GO ERET
115437          IF DILFUNIT>>3 GO ERET
115443          A:=DILGFLAG/\177700=:D; T:=PCCMBNK; *LDATX 70 % A=:FLAG WORD
115450          A/\77\/D=:DILGFLAG; X+10
115454          IF A BIT DIL1CONTROLLER THEN
115456             T:=PCCMBNK; *LDATX 0                     % A=LOG.DEV TO LOG
115460             A=:DLLOGDV; CALL LOGPH; IF A=0 GO ERET
115463          FI
115463          IF DILGFLAG BIT DIL1UNIT THEN
115466             T:=PCCMBNK; *LDATX 10                    % A=DRIVE NUMBER TO LOG
115470             IF A>>3 GO ERET; A=:DLDRIVE
115474          FI
115474          IF DILGFLAG BIT DILLIMIT THEN
115477             T:=PCCMBNK; *LDDTX 20                    % FIRST DISC ADDR TO LOG
115501             AD=:DILFADDR; *LDDTX 40                  % LAST DISC ADDR TO LOG
115503             AD=:DILGLADDR
115504          FI
115504          DILGFLAG BZERO 1DILBFULL BZERO 2DILBFULL BONE DILDEFINED=:DILGFLAG
115511  OKRET:  0=:HSTAT
115512  RETU:   IF RTRES><0 THEN CALL RTACT FI; GO WT11
115516  ERET:   -1=:HSTAT; GO RETU
115521  RBUS
115533  *"
"115533
115533
115533  %=============================================================================
115533  % 37.6          D T A P R
115533  %
115533  SUBR DTAPR
115533
115533  % TAPE READER DRIVER ON LEVEL 12
115533  DTAPR: T:=HDEV+DST; *EXR ST
115536          IF BIT 0 GO WIDENT
115540          DO
115540                IF CFREE=0 THEN CALL ID12; GO ERR22 FI
115544                TTMR=:TMR;  "DACT+DPIN"; T:=HDEV+DCONT; *EXR ST
115552  WIDENT:       CALL ID12; 0=:TMR; T:=HDEV+DDR; *EXR ST
115557                CALL RBPUT
115560                IF BHOLD>MAXBHOLD AND ISTATE><0 THEN CALL RTACT FI
115567          OD
115570
115570  RBUS
115574  *"BSIBA
"115574   %=============================================================================
115574  % 37.8          G E T D W   P U T D W
115574  %
115574  %
115574  % SPECIAL IOTRANS ROUTINES FOR SIBAS INTERNAL DEVICES
115574  % WORDS IN A,D,X,L REGISTERS WILL BE TRANSFERED
115574  %
115574  SUBR GETDW,PUTDW
115574  DISP 0; DOUBLE ARRAY POINTER DBUFST=BUFST; PSID
```

```
115574    INTEGER XREG
115575    GETDW: IF BHOLD>3 THEN
115601                 X=:XREG:=HENTE+2; AD:=DBUFST(X); *IRW ALEVB DX
115606                 A:=D; *IRW ALEVB DL
115610                 X-2; AD:=DBUFST(X)
115612                 IF X+4=MAX THEN X:=0 FI; X=:HENTE
115620                 X:=BHOLD-4=:BHOLD;=CFREE+4=:CFREE
115626                 X:=XREG; A:=:D; *IRW ALEVB DD
115631                 A:=D; L+1
115633           FI; EXIT
115634    PUTDW: T:=DFOPP=:B
115636           IF T:=CFREE-3>0 THEN
115642                 T-1=:CFREE; A=:CHARI; X=:XREG; *IRR ALEVB DD
115647                 A=:D; CHARI; AD=:DBUFST(FYLLE); *IRR ALEVB DL
115654                 A=:D=:CHARI; *IRR ALEVB DX
115657                 X+2; AD=:DBUFST(X)
115661                 IF X+2=MAX THEN X:=0 FI; X=:FYLLE
115667                 BHOLD+4=:BHOLD; X:=XREG; L+1
115674           FI; T:=DFOPP=:B; EXIT
115677    RBUS
115677    *'8BEX1
115677
115677    %=============================================================================
115677    %                        BUSCD
115677    %
115677    %       MPM4 INTERRUPT HANDLER FOR LEVEL 13D
115677    %
115677    %       ON ENTRY:                  A = IDENTCODE - 1
115677    %
115677
115677    SUBR BUSCD
115677
115677
115677    INTEGER POINTER IROUT:=ROUTI
115700    INTEGER BUSCNO,BUSST,LLIM,LPES,LPEA
115705
115705    BUSCD: *IOF
115706           A-017=:BUSCNO SHZ 2 + HDEV + 2 =:T; * IOXT          % READ BUSC STATUS
115715           A=:BUSST
115716           IF A BIT 7 GO PFINT                          % MPM4 POWER FAIL
115720           IF A BIT 6 GO MOOR                           % MEMORY OUT OF RANGE
115722           IF A BIT 5 GO PARER                          % PARITY ERROR
115724           CALL IROUT                                   % RE-INIT INTERRUPT
115725           A:=BUSCNO; T:=BUSST; CALL 9ERR(#71)          % ILLEGAL BUSC INTERRUPT
115731    RETUR: *ION
115732           CALL ID13                                    % WAIT FOR INTERRUPT
115733           GO BUSCD
115734
115734    PFINT: A:=BUSCNO; CALL 9ERR(#72); GO RETUR          % MPM4 POWER FAIL INTERRUPT
115740
115740    MOOR:  CALL IROUT
115741           A:=BUSCNO; T:=LLIM; CALL 9ERR(#73)           % MPM4 MEMORY OUT OF RANGE
115745    MOOR1: A:=LPES; T:=LPEA; CALL 9ERR(#74); GO RETUR   % WRITE PES AND PEA
115752
115752    PARER: CALL IROUT
115753           IF BUSST BIT 4 GO WRPAR
115756           A:=BUSCNO; T:=LLIM; CALL 9ERR(#75)           % MPM4 MEMORY ERROR
115762           GO MOOR1
115763
115763    WRPAR: A:=LPES SHZ -010 /\ 017 =:T                  % WRITE-PARITY ERROR
```

```
115767          A:=BUSCNO; CALL 9ERR(#76); GO RETUR
115773
115773  ROUTI:  A:=0; T:=T+1; * IOXT                    % PREPARE READ LPES
115776          T:=T-3; * IOXT                          % READ LPES
116000          A=:LPES
116001          A:=040; T:=T+3; * IOXT                  % PREPARE READ LPA
116004          T:=T-3; * IOXT                          % READ LPEA
116006          A=:LPEA
116007          A:=0106; T:=T+3; * IOXT                 % PREPARE READ LIMITS
116012          T:=T-3; * IOXT                          % READ LIMITS
116014          A /\ 0377 =:LLIM
116016          EXIT
116017
116017  RBUS
116024
116024  *"
```

```
116024
116024   %=============================================================================
116024   %                    C L O C K   I N T E R R U P T
116024   %
116024   % 37.8.2      E N T 1 3
116024   %
116024   % CLOCK INTERRUPT ENTRY, LEVEL 13
116024   INTEGER HISTFLAG=?
116024   SUBR ENT13
116024          DO
116024                  CALL ID13
116025   ENT13:         "FREQU+60001"; *IOX DCONT RTCLD
116027                  ATIME; D+1; A:=A+C; AD=:ATIME
116033                  *TRA PVL
116034                  IF A/\170=ALEVB OR =BLEVB THEN
116043                    IF BACKGROUND >< 0 THEN
116045                      MIN CURPROG.DTIN2; GO NEXT; MIN X.DTIN1;0/\0;GO NEXT
116053                    FI
116053   *"BACC
116053                     IF RTACCFLAG >< 0 THEN CALL NACCOUNT FI
116056   *"
116056                   FI
116056   NEXT:          CALL RTACT
116057   *"BHIST
116057                   IF HISTFLAG><0 THEN CALL SHISTI FI
116062   *"
116062          OD
116063   RBUS
116076
116076
116076   INTEGER ARRAY PCCS=?
116076   INTEGER IRTCPIT        % PNUMB*2+IRTCPIT=ADDR OF PROTECT AND PHYSICAL PAGE
116076                          % OF RT-COMMON ENTRY (TO BE SET INTO PIT)
116077
116077
116077   %=============================================================================
116077   %                    I N T E R N A L     I N T E R R U P T S
116077   %
116077   % 37.8.3      P O F M O N C   P O F N M O N   E F J O B
116077   %
116077   % ENTRY FOR INTERNAL INTERRUPTS, LEVEL 14
116077   %
116077
116077   *"BMCLG
116077    INTEGER CMCLG                  % MONITOR CALL LOG; =-1: LOG ALL PROGRAMS
116100                                   %                    =RT-DESCRIPTION: LOG FOR THIS PROGRAM TOO
116100   INTEGER MCLGBANK                % MEMORY BANK OF MONITOR CALL LOG TABLE
116101   INTEGER TNMCALL                 % ADDR OF TOTAL-NUMBER-OF-MON-CALL LOG TABLE
116102   INTEGER MCLGFLG                 % ><0 : MONCALL LOG STARTED
116103   INTEGER MCLGPAGE                % PHYSICAL PAGE FOR MONITOR-CALL-LOG TABLE
116104   INTEGER MCLGOWNER               % TERMINAL NUMBER OF TERMINAL ACTIVATED THE MONCALL LOG
116105   *"
116105
116105   *MGOTA=*
116105   SUBR POFMONC,POFNMON,EFJOB,FPFCOUNT
116105
116105   @ICR;
116105   INTEGER ARRAY GOTAB:=(MFELL,M1,M2,MFELL,MFELL,MFELL,MFELL,MFELL,
116105                         MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116115                         MFELL,M21,M22,M23,M24,MFELL,MFELL,MFELL,
116125                         MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116135
```

```
116145                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116155                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116165                  MFELL,MFELL,MFELL,M63,MFELL,MFELL,MFELL,MFELL,
116175                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116205                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116215                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116225                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116235                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116245                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116255                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116265                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116275                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116305                  XMSGY,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116315                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116325                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116335                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116345                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116355                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116365                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116375                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116405                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116415                  M310,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116425                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,
116435                  MFELL,MFELL,MFELL,MFELL,MFELL,MFELL,MONERR,MONERR,
116445                  MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,
116455                  MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,
116465                  MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,
116475                  MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR,MONERR);
116505  @CR;
116505  SYMBOL ERCC=15                  % ERROR CORRECTION CONTROL REGISTER
116505  SYMBOL SERC=10                  % ERROR CORRECTED BIT IN PES
116505  SYMBOL SFATAL=15,SDMA           % STATUS BITS IN N-100 PES-REGISTER
116505  DISP -200
116505          INTEGER PERR,ACTLV,IBITNO,PESERR,PEAERR,SADINSTR
116505          DOUBLE MMAP
116505  PSID
116505
116505  M1:     "INBT"; GO IOB14
116507  M2:     "OUTBT"; GO IOB14
116511  M21:    "M8INB"; GO IOB14
116513  M22:    "M8OUTB"; GO IOB14
116515  M23:    "B8INB"; GO IOB14
116517  M24:    "B8OUT"; GO IOB14
116521  M63:    "B4INW"; GO IOB14
116523  M310:   "T8INP"; GO IOB14
116525  *)FILL
116536  INTEGER FPITO:=177000,SVT14
116540  @LIB CXCPU
116540  INTEGER POINTER PT5BUPAGE:=177000+5BFPAGE+5BFPAGE
116541  INTEGER POINTER PT5UBFPAGE:=177000+5UBFPAGE+5UBFPAGE
116542  INTEGER POINTER PT3BUPAGE:=177600+5BFPAGE+5BFPAGE
116543  DOUBLE POINTER DPT5BUPAGE=PT5BUPAGE,DPT5UBFPAGE=PT5UBFPAGE,DPT3BUPAGE=PT3BUPAGE
116543  @ELIB
116543
116543  !PAGFAULT: PNUMB/\377=:PNUMB
116546
116546  @LIB CXCPU
116546          IF A=X:=5BFPAGE THEN
116551             IF RTREF.WINDOW SHZ -10=0 THEN CALL ERRFATAL FI
116556             A=:D:=162000; AD=:DPT5BUPAGE
```

```
*8SWLG
                CALL FPFCOUNT
*
                GO RET14
            FI
            IF A=X:=5UBFPAGE THEN
               IF RTREF.WINDOW/\377=0 THEN CALL ERRFATAL FI
               A=:D:=162000; AD=:DPT5UBFPAGE
*8SWLG
                CALL FPFCOUNT
*
                GO RET14
            FI
            IF A=X:=3BFPAGE THEN
               IF RTREF.WINDOW SHZ -10=0 THEN CALL ERRFATAL FI
               A=:D:=162000; AD=:DPT3BUPAGE
*8SWLG
                CALL FPFCOUNT
*
                GO RET14
            FI
@ELIB

            IF X:=BACKGROUND=0 THEN
               IF A>>=X:=ARTFPAGE AND A<<=X:=ARTLPAGE THEN      % PAGE FAULT IN RT-COMMON?
                  IF X:=DEMAND><0 THEN                          % IF DEMAND THEN TEST IF SEGMENT OVERALPS RT-CO
                     T=:SVT14
                     X:=SEGMC; CALL LIMCHECK; IF A=0 GO PFINSEGMENT
                     X:=SEGMB; CALL LIMCHECK; IF A=0 GO PFINSEGMENT
                     X:=SEGMA; CALL LIMCHECK; IF A=0 GO PFINSEGMENT
                  FI; X:=IRTCPIT; A:=PNUMB SH 1
                  X+A; A+FPITO=:T; X.DSO=:T.DSO                 % SET UP RT-COMMON IN PIT
*8SWLG
                   IF CSWLG><0 THEN
                     *MIN I (TPFR2; SKP; MIN I (TPFR1; JMP *+1
                     IF A=RTREF THEN
                        *MIN I (CPFR2; SKP; MIN I (CPFR1; JMP *+1
                     FI
                  FI
*
                   GO RET14
               FI
            FI; A:=T; GO ACTMON
MFELL:  T=:A; *IRW MLEVB DX
            "CALLPROC"
ACTMON:  *IRW MLEVB DP
            MLEV; *MST PID; MST PIE
            GO RET14

PFINSEGMENT; A:=SVT14; GO ACTMON

*8SWLG
 FPFCOUNT:                                                      % COUNT FAST PAGEFAULTS
            IF CSWLG><0 THEN                                    % SWAPPING-LOG STARTED?
               *MIN I (TFPF2; SKP; MIN I (TFPF1; JMP *+1
               IF A-RTREF=0 THEN                                % LOG FOR THIS PROGRAM?
                  *MIN I (CFPF2; SKP; MIN I (CFPF1; JMP *+1
               FI
            FI; EXIT
*
```

```
"116722    *)FILL
116757
116757    MONERR: T=:A; CALL 9ERR(#00); GO FAR ABOR
116763
116763    POFMONC: X:=377; T/\X; T=:MONNO
116766    *"8MCLG
"116766              IF X:=MCLGFLG><0 THEN                    % MONITOR CALL LOG STARTED?
116770                IF X:=CMCLG=-1 OR X=RTREF THEN         % COUNT THIS MONITOR CALL
116777                   A=:L:=MONNO SH 1+TNMCALL=:X; T:=MCLGBANK
117005                   *LDDTX; RINC DD; COPY SA ADC DA; STDTX
117011                   A:=L
117012                FI; T:=MONNO
117013             FI
117013    *"
"117013              X:=GOTAB(T); *JMP ,X; )FILL              % MONITOR CALL NUMBER IN T
117030    POFNMONC: *TRA PVL
117031           A=:D; *EXR SA
117033           A=:PERR:=D SH 11 SHZ -14=:ACTLV
117040           IF IBITNO=12 THEN CALL PPWFAIL FI
117045           IF =7 THEN PERR; T:=ACTLV; CALL 9ERR(#37); GO RET14 FI % IOX-ERROR
117055           IF =10 THEN              % MEMORY ERROR
117060                 *TRA PES
117061                 A=:T=:PESERR
117063                 IF CPSTA BIT 5N100 THEN
117066                   IF PESERR BIT 5FATAL THEN
117071                     GO NOCORR
117072                   ELSE
117073                     GO NOFATAL
117074                   FI
117074                 FI
117074                 IF PESERR BIT 5ERC THEN % 21 BITS MEMORY
117077                   A SH -11/\37+SADINSTR=:X; MMAP; *EXR SX
117105                   IF A<0 GO NOCORR         % MULTIPLE ERROR
117106    NOFATAL:       *TRA PEA
117107    NOFAP1:        A=:PEAERR
117110                   CALL 9ERR(#44); A:=0; *TRR ERCC
117114                 ELSE
117115    NOCORR:        *TRA PEA
117116    NOCP1:         A=:PEAERR; CALL 9ERR(#38)  % NOT CORRECTABLE ERROR
117121                   A:=PEAERR; T:=PESERR; CALL MFXMSG          % FATAL MEMORY ERROR IN XMSG?
117124                   IF CPSTA BIT 5N100 THEN
117127                     IF PESERR BIT 5DMA THEN CALL ERRFATAL FI  % DMA-ERROR N-100
117133                   ELSE
117134                     IF PESERR BIT 3 THEN CALL ERRFATAL FI     % DMA-ERROR N-10
117140                   FI
117140                   IF ACTLV=ALEVL GO ABOR; IF ><0 GO FAR TDTLEV
117146                 FI; GO RET14; *)FILL
117157           FI
117157           IF A=0 THEN A:=16; T:=0; CALL 9ERR(#22); GO RET14 FI  % FALSE INTERRUPT
117165           IF ACTLV=ALEVL THEN
117171                 IF IBITNO=3 THEN
117175    NOTRAP:         *TRA PGS
117176                    IF A=:PNUMB NBIT 17 THEN
117201                       *IRR ALEVB DP; AAA -1; IRW ALEVB DP
117204                    FI; T:="PPAGEFAULT"; GO FAR IPAGFAULT
117206                 FI
117206                 IF =6 THEN
117211                   RTREF.ACTPRI/\3000 SH -2\/162; X:=PERR; *TRR PCR; BSET ONE 0
117221                   CALL FINSTR; PCCS(16); *TRR PCR
117225                   IF 177600/\D=161000 THEN T:=177/\D; GO POFMONC FI
```

```
117235                ELSE IF =5 THEN; *IRR ALEVB 0; BSET ZRO SSZ DA; IRW ALEVB 0
117244                  CALL 9ERR(#30); GO ABOR; *)FILL
117263                ELSE IF =2 THEN; *TRA PGS
117270                  IF A=:T NBIT 17 THEN; *IRR ALEVB DP
117274                    A-1; *IRW ALEVB DP
117276                  FI
117276                  IF T BIT 16 THEN CALL 9ERR(#31); GO ABOR FI
117303                ELSE IF =11 THEN; *TRA PES
117310                  A=:PESERR=:T; *TRA PEA
117313                  A=:PEAERR
117314                  CALL 9ERR(#39); GO ABOR   % MEMORY OUT OF RANGE
117317                FI FI FI FI
117317                IBITNO; T:=PERR; CALL 9ERR(#24)
117323    ABOR:       "PBRTEXT"; GO FAR ACTMON
117325          FI
117325          IF =BLEVL AND IBITNO=3 AND BACKGROUND><0 THEN BLEV; *MCL PID
117340                *TRA PGS
117341                IF A=:PNUMB NBIT 17 THEN
117344                   *IRR BLEVB DP; AAA -1; IRW BLEVB DP
117347                FI; T:="P2PAGE2FAULT"; GO FAR IPAGFAULT
117351          FI
117351          IF IBITNO=3 AND ACTLV=5 THEN CALL P2XMS; GO TDTLEV; GO RET14 FI % PF IN XMSG
117364          IF IBITNO=11 THEN; *TRA PES
117371                A=:PESERR=:T; *TRA PEA
117374                A=:PEAERR; CALL 9ERR(#39)
117377          FI; GO TDTLEV; *)FILL
117411
117411    % TEST FOR ERROR ON DIRECT TASK LEVEL
117411    INTEGER CBSET(0); *BSET ONE DA
117412    TDTLEV: IF ACTLV<6 THEN
117416                IF A=2 GO DTLERR
117421                CALL ERRFATAL
117422          FI
117422          IF A>11 AND A><17 THEN CALL ERRFATAL FI
117431    DTLERR: A SH 3\/CBSET=:T; A:=0; *EXR ST; MCL PID; MCL PIE
117440          T:=IBITNO; ACTLV; CALL 9ERR(#04); *TRA PGS
117445          GO RET14
117446
117446    *)FILL
117451
117451    *"88EX1
117451     RBUS
117451
117451
117451    %=============================================================================
117451    %              N W 9 E R R
117451    %
117451    % PART OF THE 9ERR ROUTINE
117451    %
117451    % ENTRY:       A=ERPREG
117451    %              D=N1
117451    %              X=ERNUM
117451    %
117451    SUBR NW9ERR
117451    INTEGER ARRAY ERRARR(0)
117451    INTEGER ERNUM,ERPREG,N1,N2,RTPROG
117456    INTEGER POINTER LREG
117457    INTEGER BREG
117460
117460    NW9ERR: A=:ERPREG:=D=:N1; X=:ERNUM; T=:N2:=B=:BREG:=L=:"LREG"
```

```
117471          IF X=#29 THEN 9ESSAVTAD; A=:ERPREG; 9EXREG=:N1 ELSE T:=RTREF FI; T=:RTPROG
117503          IF ERNUM=#90 OR =#91 THEN 9EDREG=:ERPREG FI
117514          IF RTPROG=0 THEN GO ASRT FI                     % IF ERROR DURING STARTUP
117517          IF A.ACTPRI BIT 17 AND ERNUM><#20 AND><#21 AND><#22 AND><#23 AND><#37
117541             AND ><#40 AND ><#44 AND ><#46 AND ><#47 AND><#72 AND><#73 AND><#74
117566             AND ><#75 AND ><#76 AND ><#97 AND ><#99 AND ><-1 AND >#08 OR <#06 THEN
117615                X:="ERNUM"; CALL BGERR
117617          ELSE
117620    ASRT:                                                 % RT-ERROR
117622                "IERRFIELD"=:B
117626                IF CFREE>=5 THEN
117637                  FOR X:=0 TO 4 DO ERRARR(X)=:9ERRP(X); CALL RWPUT OD
117640                  CALL XRTACT
117640                FI
117643          FI; BREG=:B; GO LREG
117702    RBUS
117702
117702    %=============================================================================
117702    %              R E S Y S   -   C O L D S T A R T
117702    %
117702    SUBR RESYS,COLDSTART
117702
117702    INTEGER CPGN=?,CCPGN=?
117702    INTEGER ARRAY CDSKFORM(10)
117712
117712    % FIND DISC-LAYOUT OF MAIN DISK
117712    FCDSKFORM: IF "TRNSF"="BDISK" THEN
117716                  X:="HTABL"+B; X.SO; X:="CDSKFORM"; T:=20=:D; *MOVB; JMP *
117726                  "QDLAY" SHZ -12=:CPGN
117731               FI; EXIT
117732
117732    INTEGER ROUTSWITCH,CPGN,CCPGN
117735    INTEGER CBLDA(0); *BLDA 00 DT
117736
117736    % ENTRY WITH A=LOGICAL DEVICE NUMBER TO COMMUNICATE WITH AFTER COLD-START
117736    COLDSTART: *IOF
117737          A=:ROUTSWITCH; GO FELLS
117741    RESYS: *IOF
117742          0=:ROUTSWITCH
117743          GO FELLS
117744    *)FILL
117750
117750    INTEGER CNBLCK,CCOUNT
117752    DOUBLE CMASS,DMASS,CCMAD(0); *0;2000
117760    % GET BOOTSTRAP;
117760
117760    FELLS:
117760    L1:     MASSNO(0); CALL LOGPH; IF =0 GO ERRF; A=:B
117765          ABLPAGE(X)=:CNBLCK
117767
117767    ERR2:   IF ROUTSWITCH><0 THEN
117771             CALL FCDSKFORM; 0=:CCPGN
117773             -77=:CCOUNT; DDASA=:CMASS; DDBLST=:DMASS
120001             FOR CCOUNT DO
120001                AD:=CCMAD=:MEMAD=:CMADR
120004    LOOPX:      T:=DSKTYP BZERO "0"; CMASS; X:=CNBLCK
120010    RLOOP:      CALL TRNSF; GO ERR2; GO RLOOP
120013                IF CCOUNT=-77 THEN
120017                   AD:=CCMAD; X:="LGCOLDSTART"+D; ROUTSWITCH=:X.SO
120024                FI
120024                IF CPGN><0 AND A=CCPGN THEN
```

```
20031                       AD:=CCMAD; A:="QDLAY"/\1777; D+A=:X
20036                       A:="CDSKFORM"; T:=20=:D; *MOVB; JMP *
20043                    FI; AD:=CCMAD=:MEMAD=:CMADR
20046                    T:=DSKTYP BONE "0"; DMASS; X:=CNBLCK
20052    WLOOP:          CALL TRNSF; GO ERR2; GO WLOOP
20055                    CMASS; A:=:D+CNBLCK; D:=D+C; A:=:D; AD=:CMASS
20063                    DMASS; A:=:D+CNBLCK; D:=D+C; A:=:D; AD=:DMASS
20071                    MIN CCPGN
20072                 OD
20074              FI
20074              A:=0=:D; AD=:MEMAD=:CMADR; T:=DSKTYP; X:=CNBLCK
120102             *TRR 10
120103   LOOP2:    (ALL TRNSF; GO ERR2; GO LOOP2
120106             "0"; *TRR IIE; IDENT PL11; IDENT PL11
120112             A:=0; *TRR 11          % LOWER LIMIT
120114             A:=-1; *TRR 12         % UPPER LIMIT
120116             *TRR 10                % CLEAR CACHE
120117             0=:P                   % START BOOTSTRAP
120120   ERRF:     *IOF; WAIT
120122   RBUS
120134
120134   $DEV 1
120134   $DEV (S-S-J)MRES-SEGADM
```

```
120134
120134   %=============================================================================
120134   %
120134   %   3 7 . 0      S E G M E N T    S U P E R V I S I N G
120134   %
120134   %=============================================================================
120134
120134   *'8SWLG
"120134   %=============================================================================
120134   %      S W A P P I N G  -  L O G  D A T A
120134   %
120134   INTEGER CSWLG                  % ><0 : SWAPPING-LOG STARTED
120135                                  % = RT-DESC. ADDR: LOG FOR A SPESIFIC PROGRAM TOO.
120135   INTEGER CCSWLG                 % SAVED CSWLG
120136   INTEGER SWLGOWNER              % TERMINAL NUMBER OF USER ACTIVATED SWAPPING LOG
120137   INTEGER TFPF1,TFPF2            % TOTAL FAST PAGE FAULTS (REQUIRING NO DISC ACCESS)
120141   DOUBLE TFPFS=TFPF1
120141   INTEGER CFPF1,CFPF2            % NUMBER OF FAST PAGE FAULTS FOR A SPESIFIC PROGRAM
120143   DOUBLE CFPFS=CFPF1
120143   INTEGER TPFR1,TPFR2            % TOTAL NUMBER OF PAGE FAULTS IN RT-COMMON
120145   DOUBLE TPFRS=TPFR1
120145   INTEGER CPFR1,CPFR2            % NUMBER OF PAGE FAULTS IN RT-COMMON FOR A SPESIFIC PROGRAM
120147   DOUBLE CPFRS=CPFR1
120147   INTEGER TPF41,TPF42            % TOTAL NUMBER OF PAGE FAULTS ON LEVEL 4
120151   DOUBLE TPFL4=TPF41
120151   INTEGER CPF41,CPF42            % NUMBER OF PAGE FAULTS ON LEVEL 4 FOR A SPESIFIC PROGRAM
120153   DOUBLE CPFL4=CPF41
120153   INTEGER TPF11,TPF12            % TOTAL NUMBER OF PAGE FAULTS ON LEVEL 1
120155   DOUBLE TPFL1=TPF11
120155   INTEGER CPF11,CPF12            % NUMBER OF PAGE FAULTS ON LEVEL 1 FOR A SPESIFIC PROGRAM
120157   DOUBLE CPFL1=CPF11
120157   INTEGER SNWP1,SNWP2            % NUMBER OF PAGES SWAPPED OUT
120161   DOUBLE SNWPS=SNWP1
120161   *"
"120161
120161   INTEGER FIXCRT                         % RT-PROGRAM EXECUTING FIXC
120162
120162
120162   %=============================================================================
120162   % 37.1        S E G A D M
120162   %
120162   % SUBROUTINE TO CHECK IF SEGMENTS ARE OK, CALLED FROM STUPR
120162   % RETURN IF SEGMENTS OK, ELSE TO SEGIN
120162   % ENTRY:      A=ACTSEG, X=RTREF
120162
120162   SUBR SEGADM
120162   INTEGER POINTER LREG; INTEGER BREG
120164   INTEGER XREG=?
120164
120164   SEGADM: IF X=FIXCRT GO XCSEGS
120167           T:=L=:"LREG":=B=:BREG; X=:XREG
120174           A=:D SHZ -10; IF ><0 THEN A*5SEGSIZE+SEGSTART FI; A=:NSEGA
120202           377/\D;       IF ><0 THEN A*5SEGSIZE+SEGSTART FI; A=:NSEGB
120210           X.RSEGM/\377; IF ><0 THEN A*5SEGSIZE+SEGSTART FI; A=:NSEGC
120216   SLOWS:  NSEGA=:B; CALL SEGCHECK; IF A<0 THEN CALL SEGIN FI
120223           NSEGB=:B; CALL SEGCHECK; IF A<0 THEN CALL SEGIN FI
120230           NSEGC=:B; CALL SEGCHECK; IF A<0 THEN CALL SEGIN FI
120235           CALL XCSEGS            % CLEAR PAGETABLES
120236           NSEGC=:SEGMC; CALL STSEG; NSEGC; X:=RTREF; CALL CLNREENT
120244           NSEGA=:SEGMA; CALL STSEG; NSEGB=:SEGMB; CALL STSEG
```

```
 20252   FILLS: XREG=:SEGPROG
 20254          A:=0; IF X:=SEGMA><0 THEN X.FLAG FI
 20260          IF X:=SEGMB><0 THEN A\/X.FLAG FI
 20263          IF A NBIT 5SYSEGM THEN CALL SETRT FI
*20266  *"BLAMU
 20266           CALL SLAMU
*20267  *"
 20267   OUT:   BREG=:B; X:=XREG; GO LREG
 20273  INTEGER XREG
 20274  *)FILL
 20317
 20317  RBUS
 20317
 20317  %-------------------------------------------------------------------------------
 20317  % 37.2        P A G E F A U L T   P A G E 2 F A U L T   L I M C H E C K
 20317  %
 20317  %--MONITOR ENTRY FROM LEVEL 14 IF PAGE FAULT
 20317  % A=PAGE NUMBER FOUND ON LEVEL 14
 20317  %EXIT TO SEGIN
 20317
 20317  SUBR PAGEFAULT,PAGE2FAULT,LIMCHECK
 20317
 20317  DOUBLE ARRAY POINTER WINDAD:=177174
 20320  DOUBLE POINTER BPG:=177000+5BFPAGE+5BFPAGE
 20321  INTEGER SHAINSTR(0); *SHA
 20322  INTEGER CINSTR(0); *BSKP ZRO DT
 20323
 20323  *"BSWLG
*20323   INTEGER SWLGF
 20324  *"
*20324
 20324  PAGE2FAULT:
 20324  *"BSWLG
*20324                 0=:SWLGF
 20325  *"
*20325                 CALL BLEVSET; GO PAGF       % PAGEFAULT ON INBT/OUTBT LEVEL
 20327  PAGEFAULT:                                 % PAGEFAULT ON APPLICATION LEVEL
 20327  *"BSWLG
*20327                 1=:SWLGF
 20331  *"
*20331   PAGF:  IF PNUMB=76 OR =77 THEN
 20340  @LIB CXCPU
 20340             CALL ERRFATAL
 20341  @ELIB
 20341  @LIB CXCPU-,
 20341          ELSE
 20342             IF A="5BFPAGE" THEN
 20345  @LIB CXCPU
 20345               CALL ERRFATAL
 20346  @ELIB
 20346  @LIB CXCPU-,
 20346             FI
 20346          FI
 20346          PNUMB SH 1+177000=:X
 20352          IF X.S0><0 THEN CALL ERRFATAL FI
 20355          IF DEMAND=0 THEN PNUMB; CALL 9ERR(#13); GO PBRTEXT FI
 20363          X:=SEGMC; CALL LIMCHECK
 20365          IF =0 THEN
 20366                A:=X.LOGADR/\300=:D:=PNUMB/\377-D
 20374                AD SHZ -4; X:=RTREF.RTDLGADDR+A
```

Is page table entry = zero ?

```
120400                  SHAINSTR; D SHZ -14+A; 1; *EXR SD
120405                  IF A/\X.BITMAP><0 GO NOTREENT; X:=SEGMC
120410          ELSE
120411  NOTREENT:       X:=SEGMA; CALL LIMCHECK
120413                  IF <0 THEN X:=SEGMB; CALL LIMCHECK
120416                     IF <0 THEN
120417  OUTSIDE:              PNUMB; CALL 9ERR(#14); GO PBRTEXT
120423                     FI
120423          FI      FI; X=:B; CALL SEGIN
120425  *)FILL
120444
120444  LIMCHECK: IF X=0 OR X.LOGADR/\377>PNUMB OR A=:D:=X.LOGADR SHZ -10+D<=T THEN
120460                  A:=-1
120461               ELSE
120462  *"8SWLG
"120462                  IF CSWLGF><0 THEN                                % SWAPPING-LOG?
120464                     IF T:=SWLGF=0 THEN                           % LEVEL 4
120467                        *MIN I (TPF42; SKP; MIN I (TPF41; JMP *+1
120473                        IF A=RTREF THEN                           % LOG FOR A SPESIFIC PROGRAM
120476                           *MIN I (CPF42; SKP; MIN I (CPF41; JMP *+1
120502                        FI
120502                     ELSE
120503                        *MIN I (TPF12; SKP; MIN I (TPF11; JMP *+1 % LEVEL 1
120507                        IF A=RTREF THEN                           % LOG FOR A SPESIFIC PROGRAM
120512                           *MIN I (CPF12; SKP; MIN I (CPF11; JMP *+1
120516                        FI
120516                     FI;
120516                  FI
120516  *"
"120516              A:=0                                              % RETURN WITH A=0
120517              FI; EXIT
120520  RBUS
120534
120534  %=============================================================================
120534  % 37.3           C L S E G   S T S E G   X C L S E G   Y C L S E G
120534  %               C L P A G E   S L A M U   C L A M U
120534  %
120534  %  SUBROUTINES TO UPDATE THE HARDWARE PAGE TABLE
120534  % ENTRY:        A=SEGM. POINTER
120534
120534  SUBR CLSEG,STSEG,CLPAGE,SCPOUT,CLWINDOWS,CLAMU,SLAMU
120534  *"8LAMU
"120534   DOUBLE ARRAY POINTER PTBGO=?
120534  INTEGER ARRAY POINTER IPTBG=?
120534  INTEGER POINTER LREG
120535  INTEGER CL,CACLE
120537
120537  SLAMU:  K:="0"; GO LAMU
120541  CLAMU:  K:=1
120542  LAMU:   A:=L=:"LREG"; IF A:=SEGPROG = 0 GO LREG
120547          A-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
120554          A*GNLPRT*ALMSZ+LAMACT=:CACLE; 0=:CL
120561          FOR CL TO GNLPRT-1 DO
120566                  A*ALMSZ+CACLE=:X; T:=LAMBANK; *LDDTX
120573                  IF A >< 0 THEN              % D=FIRST LOGICAL PAGE NO
120574                     A*LDTSZ+LAMDT=:X; *LDATX LMNP
120600                     L:=D+A-1+L; *LDATX LMPP;
120605                     * LDXTX LMPR; COPY SX DT  % T= LAMU PROTECTION
120607                     X:=D+X; D:=A; A:=T
120613                     IF K THEN
```

```
120615                      FOR X STEP 2 TO L DO 0=:IPTBG(X) OD
120622                   ELSE
120623                      FOR X STEP 2 TO L DO AD=:PTBG0(X); D+1 OD
120631                   FI
120631                FI
120631          OD; GO LREG
120636
120636   *)FILL
120646   *
120646
120646   INTEGER PINDEX,BRG
120650   DOUBLE POINTER P76:=177174,P77:=177176
120652   INTEGER POINTER IP76=P76,IP77=P77
120652   DOUBLE ARRAY POINTER PTBG0:=177000,PTBG1:=177002
120654   INTEGER ARRAY POINTER IPTBG=PTBG0
120654   INTEGER POINTER BPG:=177000+5BFPAGE+5BFPAGE
120655   * CXCPU
120655    INTEGER POINTER PT5UBFPAGE:=177000+5UBFPAGE+5UBFPAGE
120656   *
120656    CLWINDOWS;
120656   @LIB CXCPU
120656                0=:PT5UBFPAGE
120657   @ELIB
120657   @LIB CXCPU-,
120657                0=:BPG; EXIT
120661
120661   CLSEG: IF A=0 THEN EXIT FI
120663          X=:PINDEX; A:=:B; A=:BRG
120666   @LIB CXCPU-,
120666   L1:    T:=CORMBANK; X:=BPAGLINK
120670   @LIB CXCPU
120670          *ICLEP; JMP OUT
120672   @ELIB
120672   @LIB CXCPU-,
120672
120672   STSEG: IF A=0 THEN EXIT FI
120674          X=:PINDEX; A:=:B; A=:BRG
120677          T:=CORMBANK; X:=BPAGLINK
120701   @LIB CXCPU
120701          *ISETP
120702   @ELIB
120702   @LIB CXCPU-,
120702   SCPOUT:
120702   OUT:   X:=BRG=:B:=PINDEX
120705          EXIT
120706   CLPAGE: T:=B=:BRG; T:=CORMBANK; *LDBTX DALOG; 177700   % MIC. PROG. BUG!!!
120713          0=:S0; BRG=:B; EXIT
120717   RBUS
120720
120720   %=========================================================================
120720   %  37.5         S E T R T    C L E R T
120720   %
120720   %  ROUTINES FOR SETTING/CLEARING RT'S SYSTEM SEGMENT PIT ENTRIES
120720
120720   SUBR CLERT,SETRT
120720
120720   INTEGER XREG,BREG
120722   INTEGER ARRAY POINTER CPITADDR:=177000+SYSST+2+SYSST+2
120723   DOUBLE ARRAY POINTER DCPITADDR=CPITADDR
120723
```

```
120723   CLERT: X=:XREG:=B=:BREG
120726          FOR X:=0 STEP 2 TO "5SSSZ-5ESSZ-1" SH 1 DO 0=:CPITADDR(X) OD
120736          CCFPAGE SH 1+177200=:X; T:=177400
120743          DO WHILE X<<T; 0=:X.S0; X+2; 0D
120750          GO OUT
120751
120751   SETRT: A:=B=:BREG; X=:XREG; T:=0
120755          A:="SYSST+2"=:D:=162000
120760          FOR X:=0 STEP 2 TO "5SSSZ-5ESSZ-1" SH 1 DO AD=:DCPITADDR(X); D+1; OD
120771   OUT:   X:=BREG=:B:=XREG; EXIT
120775   RBUS
121003
121003
121003   %-----------------------------------------------------------------------------
121003   % 37.6        S E G C H E C K
121003   %
121003   % SUBROUTINE TO CHECK IF A SEGMENT IS OK
121003   % B=ADDR. IN SEGMENT TABLE; RETURN: A=0 IF OK,ELSE -1
121003   % IF OK THE SEGMENT IS PLACED IN START OF SEGMENT LIST
121003
121003   SUBR SEGCHECK
121003   INTEGER XREG
121004   SEGCHECK: X=:XREG; IF B=0 GO OK
121007          IF FLAG BIT 5OK THEN
121012                IF A BIT 5DEMAND THEN T:=1=:DEMAND FI
121016                IF SEGLINK><0 AND BSEGLINK><B THEN
121023   LOOP:          IF A.SEGLINK=B THEN SEGLINK=:X.SEGLINK  %REMOVE
121031                  ELSE IF ><-1 GO LOOP FI
121035                  BSEGLINK=:SEGLINK; A:=B=:BSEGLINK       %INSERT
121041                FI
121041   OK:          A:=0
121042          ELSE -1
121044          FI
121044          X:=XREG; EXIT
121046   RBUS
121050
121050   %-----------------------------------------------------------------------------
121050   % 37.7        C L N R E E N T R
121050   %
121050   % SUBROUTINE TO RESET NONREENTRANT PAGES IN PAGE INDEX TABLE
121050   % A=SEGMENT ADDRESS,X=RT-DESCR
121050   SUBR CLNREENTR
121050   @LIB CXCPU
121050   CLNREENTR: X=:D:=X.RTDLGADDR+5BITMAP-25; T:=34; *ICLNR
121056              X:=D; EXIT
121060   @ELIB
121060   @LIB CXCPU-,
121060   RBUS
121060
121060   %-----------------------------------------------------------------------------
121060   % 37.8        C S E G S   X C S E G S   C H R E E N T P A G E S
121060   %
121060   % SUBROUTINES TO CLEAR SEGMENTS
121060   % CSEGS CLEARS IF SEGMENT IS PRESENT (A=SEGMENT POINTER)
121060   % XCSEGS CLEARS ALWAYS
121060
121060   SUBR CSEGS,XCSEGS,CHREENTPAGES
121060   INTEGER XREG
121061   INTEGER POINTER IPIT3:=177600+5BFPAGE+5BFPAGE
121062   INTEGER POINTER IPIT0:=177000+5BFPAGE+5BFPAGE
```

```
21063   INTEGER POINTER LREG,UPITO:=177000+5UBFPAGE+5UBFPAGE
21065   CSEGS: IF A><SEGMA AND ><SEGMB AND ><SEGMC THEN EXIT FI
21077   XCSEGS: X=:XREG:=L=:"LREG"
21102   @LIB CXCPU
21102           0=:IPITO=:IPIT3=:UPITO
21105   @ELIB
21105   *"8LAMU
121105           CALL CLAMU
21106   *"
121106           SEGMA; CALL CLSEG; SEGMB; CALL CLSEG; SEGMC; CALL CLSEG
21114           CALL CLERT              % CLEAR PIT ENTRIES FOR RT'S SYSTEM SEGMENT
21115           IF SEGMC><0 THEN CALL CHREENTPAGES FI
21120           0=:SEGMA=:SEGMB=:SEGMC
21123           X:=XREG; GO LREG
21125   *)FILL
21134
21134   DOUBLE SVAD
21136   INTEGER CSEG
121137  NEWLOOP: X:=CSEG+1; D:=0; T:=CORMBANK
21143           DO
21143              *ICHRE
121144             GO WIPSET; GO FAR OUT
121146  WIPSET:    AD=:SVAD; CALL COUTLINK
121150             AD:=SVAD; A=:X
121152          OD
121153
121153  INTEGER CALOGNO,EALOGNO,ERINDICATOR
121156  INTEGER POINTER LREG2
121157  INTEGER PREVX,PREVT,CURX,BRG,SHAINSTR(0); *SHA
121164  CHREENTPAGES: A=:CSEG; IF A.FLAG NBIT 5WPM THEN EXIT FI
121172          A:=L=:"LREG2"; A:=B=:BRG; 0=:ERINDICATOR
121177  @LIB CXCPU
121177          GO NEWLOOP
121200  @ELIB
121200  @LIB CXCPU-,
121200  OUT:    BRG=:B; IF CSEG.BPAGLINK=0 THEN CALL SREMOVE FI
121206          IF ERINDICATOR><0 GO PSBRTEXT       % ABORT PROGRAM
121211          GO LREG2
121212
121212  INTEGER POINTER LREG3
121213  COUTLINK: A:=L=:"LREG3"; X=:CURX
121216          SEGMA; CALL INSRPAGE                % INSERT PAGE?
121220          IF A=0 THEN SEGMB; CALL INSRPAGE    % INSERT PAGE?
121223             IF A=0 THEN                      % OUTSIDE SEGMENT BOUNDS
121224                *LDATX DALOG
121225                A=:EALOGNO
121226                IF "XSGRT".SEGLINK=0 THEN     % SET DUMMY LAST IN SEGMENT LINK
121231                   X:="BSEGLINK"
121232                   DO WHILE X.SEGLINK><-1; X:=A; OD
121240                   "XSGRT"=:X.SEGLINK; -1=:"XSGRT".SEGLINK
121245                FI; "XSGRT"=:B; A:=BPAGLINK; X:=CURX=:BPAGLINK
121252                T:=CORMBANK; *STATX DPAGL          % LINK PAGE TO DUMMY
121254                A:=1; *STATX DPGPR
121256                RTREF=:CALOGNO; SEGPROG=:RTREF=:ERINDICATOR
121263                EALOGNO; CALL 9ERR(#14); CALOGNO=:RTREF
121270                GO LREG3
121271             FI
121271          FI; *LDDTX DPAGL
121272          CSEG.LOGADR/\300; D-A; A:=D; AD SHZ -4
121300          X:=SEGPROG.RTDLGADDR+A
```

```
121303          SHAINSTR; D SHZ -14+A; 1; *EXR SD
121310          A\/X.BITMAP=:X.BITMAP; GO LREG3
121313
121313    RBUS
121331
121331    % ============================================================================
121331    % 37.9          I N S R P A G E
121331    %
121331    % SUBROUTINE TO INSERT A PAGE INTO A SEGMENT
121331    % A=SEGMENT ADDRESS, X=CORMAP ELEMENT ADDRESS
121331    % RETURN: A=SEGMENT NUMBER IF OK, ELSE A=0
121331
121331    SUBR INSRPAGE
121331    INTEGER BREG
121332    INSRPAGE: IF =0 THEN EXIT FI
121334          A:=:B=:BREG; IF FLAG NBIT 5DEMAND THEN A:=0; GO OUT FI
121343          *LDATX DALOG
121344          A=:T; LOGADR; AD SHZ -10; D SHZ -10
121350          IF T>=D AND T<D+A THEN                          % INSIDE, INSERT
121355             A:=BPAGLINK; T:=CORMBANK; *STATX DPAGL
121360             A:=B-SEGSTART=:D:=0; T:=5SEGSIZE; *RDIV ST
121366             T:=CORMBANK; A=:D; *LDATX DPGPR
121371             A/\177400\/D; *STATX DPGPR
121374             X=:BPAGLINK
121375             IF SEGLINK=0 THEN BSEGLINK=:SEGLINK; A:=B=:BSEGLINK FI
121403             A:=D                  % OK: A=SEGMENT NUMBER
121404          ELSE
121405             A:=0                  % NOT OK
121406          FI
121406    OUT:  T:=BREG=:B:=CORMBANK
121411          EXIT
121412    RBUS
121416
121416
121416
```

```
121416
121416    % ============================================================================
121416    %
121416    %                  S E G M E N T   F E T C H I N G
121416    %
121416    % ============================================================================
121416    % 37.10       S E G I N
121416
121416    % SUBROUTINE TO GET A NEW SEGMENT INTO CORE
121416    % B=SEGMENT ELEMENT; NO RETURN
121416
121416    SUBR SEGIN
121416    SYMBOL 5READ,5WRITE
121416    SEGIN: 5CLOAD; X:=RTREF; CALL RESERVE
121421           IF <0 THEN                          %SEG. TRANSFER GOING ON
121422              IF SWPFLAG><0 THEN
121424                 X.ACTPRI BONE 5SWWAIT=:X.ACTPRI
121427                 X.STATUS BONE 5WAIT=:X.STATUS
121432    LOOP:        RTREF+"BRESLINK"=:X
121435                 DO WHILE X:=X.RESLINK><RTREF
121441                    IF X.TYPRING/\3=3 THEN X=:B:=RTREF; CALL BRELEASE; GO LOOP FI
121452                 OD
121453              FI; GO PRWAIT; *)FILL
121464           FI; A:=B=:SEGREF
121466           X=:SRTREF; NSEGA=:TSEGA; NSEGB=:TSEGB; NSEGC=:TSEGC
121475           CALL XCSEGS
121476           IF FLAG NBIT 5DEMAND THEN
121501
121501    %NON-DEMAND SEGMENT
121501              DO                    %UNTIL THE SEGMENT HAS GOT ENOUGH PAGES
121501                 IF B="XSGRT" THEN -1=:NUMBER; GO XLRU FI       % DUMMY SEGMENT
121507                 LOGADR SHZ-10-=:NUMBER
121513                 IF X:=BPAGLINK><0 THEN
121515                    LOGADR/\377=:D; T:=CORMBANK; *LDATX DALOG
121522                    D-A=:A; A=:NUMBER
121525                 FI
121525              WHILE NUMBER><0     %ENOUGH PAGES
121527    XLRU:        CALL LRU; X:=:B; FLAG =:SAVEFLAG BZERO 5OK=:FLAG
121535                 5WRITE; CALL TRNSEG
121537                 IF ><0 THEN SAVEFLAG=:FLAG; GO END1 FI
121543                 X:=:B; CALL LINKOVER
121545              OD
121546           ELSE
121547
121547    %DEMAND SEGMENT
121547              IF FLAG BIT 5OK THEN
121552                 -1=:NUMBER; PNUMB=:CPNUMB
121556                 FLAG BZERO 5OK=:FLAG
121561                 CALL LRU; X:=:B; FLAG BZERO 5OK=:FLAG
121566                 5WRITE; CALL TRNSEG; X:=:B; CALL LINKOVER
121572                 T:=CORMBANK; X:=BPAGLINK; *LDATX DPAGL
121575                 IF A><0 THEN
121576                    A=:BPAGLINK; *STZTX DPAGL
121600                    X=:D
121601                    DO WHILE A><0
121602                       A=:X; *LDATX DPAGL
121604                    OD; A:=D; *STATX DPAGL
121607                 FI
121607              FI
121607           FI
```

```
121607          U=:NUMBER; 5READ; CALL TRNSEG
121612  %SEGMENT OK:
121612          FLAG BONE 50K=:FLAG
121615  END1:   X:=SRTREF; 5CLOAD; CALL RELEASE
121620          IF X=RTREF GO PRW; GO PSTUPR
121624  RBUS
121654
121654  %-----------------------------------------------------------------------------
121654  % 37.11        L R U
121654
121654  %SUBROUTINE TO FIND LEAST RECENTLY USED SEGMENT
121654  %ENTRY: B=SEGREF
121654  % RETURN: X=LRU SEGMENT POINTER
121654
121654  SUBR LRU
121654  INTEGER ARRAY TSGN:=(TSEGB,TSEGC,TSEGA,TSEGC,TSEGA,TSEGB)
121662  INTEGER POINTER LREG; INTEGER LRUS2,R01
121665  LRU:    A:=L=:"LREG"
121667          IF MAXP<< 200 AND FLAG BIT 5DEMAND THEN             % COUNT PAGES
121676             X:=BPAGLINK; D:=0; T:=CORMBANK
121701             DO WHILE X><0
121702                *LDXTX DPAGL
121703                D+1
121704             OD
121705             IF D>=MAXP THEN B=:X; GO OUT FI
121712          FI
121712          FOR X:=-3 DO IF TSEGS(X)=0 GO NOTF
121715                X:=:A; IF X:=X.SEGLINK=0 GO NOTF; A=:X
121721          OD; CALL GETLAST; IF =0 GO OUT
121724  % ALL THREE SEGMENTS PRESENT, TAKE ONE OF THE TWO OTHERS
121724          FOR X:=-3 DO WHILE TSEGS(X)><B OD
121731          R01-,=:R01; X+3+X-A; X:=TSGN(X); X:=X.S0; GO OUT
121742  NOTF:   CALL GETLAST; IF =0 GO OUT
121744          CALL GETLAST; X=:LRUS2
121746          IF <0 THEN CALL GETLAST
121750                IF <0 THEN
121751                  X=:D; 2/\X.FLAG; T:=LRUS2.FLAG; *BLDA 10 DT; BSTA 0 DA
121760                  D=:X; GOSW ERR,FRSTDEM,OUT,BOTHDEM
121766  BOTHDEM:        IF X><B GO OUT
121770  FRSTDEM:        X:=LRUS2
121771          FI    FI
121771  % X=SEGMENT TO BE REMOVED(LRU-SEG):
121771  OUT:    X=:A; CALL CSEGS
121773          IF X.SEGLINK=0 GO NOTF
121775          X:=:B; CALL SEGSORT; CALL RANDOM; X:=:B
122001          GO LREG
122002  ERR:    IF SEGREF="XSGRT" THEN     % SEGMENT 1, FIXC
122006             "CLFIE"=:B; X:=SRTREF; CALL BRELEASE; 0=:X.ACTSEG; GO PSTUPR
122014          FI
122014          SRTREF=:RTREF; CALL 9ERR(#19); GO PRTEXT
122021  *)FILL
122043
122043  % LOCAL SUBROUTINE TO FIND LAST SEGMENT
122043  % RETURN: X=LRU-SEG;IF =TSEGA OR TSEGB OR =TSEGC :A<0
122043  INTEGER PREVLINK
122044  GETLAST: "BSEGLINK".SEGLINK; IF =-1 GO ERR
122051          DO X=:PREVLINK WHILE A.SEGLINK><-1; OD
122060          X=:D; A:=0
122062          IF X=TSEGA OR X=TSEGB OR X=TSEGC THEN
122073                IF BSEGLINK><X THEN
```

```
122076                  A=:X.SEGLINK; X=:BSEGLINK; -1=:PREVLINK.SEGLINK
122103                FI; -1; D=:X
122105          FI; EXIT
122106    RBUS
122112
122112    %-------------------------------------------------------------------------
122112    % 37.12        S E G S O R T
122112
122112    % SUBROUTINE TO PUT NON-USED PAGES FIRST IN PAGE LINK
122112    % B=SEGMENT
122112    SUBR SEGSORT
122112    INTEGER LST0,LST0BANK,LST1,LST1BANK,FRST1,XREG
122120    SEGSORT: IF FLAG NBIT 5DEMAND THEN EXIT FI;
122124          X=:XREG;=BPAGLINK; A:=B+1=:LST0; 0=:LST0BANK; "FRST1"=:LST1
122133          0=:LST1BANK
122134          DO
122134             T:=CORMBANK; *LDATX DPGPR
122136             A:=:X
122137             IF X BIT 5PGU THEN
122141                T:=LST1BANK; X:=LST1; *STATX
122144                A=:LST1; T:=CORMBANK=:LST1BANK
122147             ELSE
122150                T:=LST0BANK; X:=LST0; *STATX
122153                A=:LST0; T:=CORMBANK=:LST0BANK
122156             FI; X:=A; *LDXTX DPAGL
122160          WHILE X><0
122161          OD; X:=LST1; T:=LST1BANK; *STZTX
122165          FRST1; X:=LST0; T:=LST0BANK; *STATX
122171          X:=XREG; EXIT
122173    RBUS
122175
122175    %-------------------------------------------------------------------------
122175    % 37.13        R A N D O M
122175
122175    % SUBROUTINE TO TAKE A RANDOM(PREFERABLY NOT USED) PAGE AND PUT IT FIRST
122175    % ENTRY:       B=SEGMENT
122175    SUBR RANDOM
122175    INTEGER XREG,XN:=3614,CC:=33031,AA:=12465,PREVX,PREVT
122203    RANDOM: IF FLAG NBIT 5DEMAND THEN EXIT FI
122207          X=:XREG; X:=BPAGLINK; D:=0; T:=CORMBANK
122213          DO
122213             IF X=0 GO L1; *LDATX DPGPR
122215             IF A BIT 5PGU GO L1
122217             D+1; *LDXTX DPAGL
122221          OD
122222    L1:   IF D=0 THEN
122224             DO WHILE X><0
122225                D+1; *LDXTX DPAGL
122227             OD
122230          FI
122230          D=:T; XN*AA+CC=:XN BZERO 17=:D:=0; *RDIV ST
122241          X:=BPAGLINK; A:=B+1=:PREVX; 0=:PREVT; T:=CORMBANK
122246          FOR D-+1 TO 0 DO
122252             X=:PREVX; T=:PREVT; *LDXTX DPAGL
122255          OD; *LDATX DPAGL
122260          X=:T:=PREVX; T=:PREVX:=PREVT; *STATX DPAGL
122265          X:=PREVX; T:=CORMBANK; BPAGLINK; *STATX DPAGL
122271          X=:BPAGLINK
122272          X:=XREG; EXIT
122274    RBUS
```

```
122275
122275   %=============================================================================
122275   % 37.14       T R N S E G  R W S E G M  X T R N S E G
122275
122275   %SUBROUTINE TO READ OR WRITE PAGES OF A SEGMENT; NUMBER=-NEEDED PAGES
122275   % ENTRY:      A=READ/WRITE FLAG, B=SEGMENT
122275   %RETURN WITH A=0 IF NO TRANSFER HAS BEEN DONE
122275
122275   SUBR TRNSEG,RWSEGM,XTRNSEG,INRWSEGM
122275
122275   INTEGER WRITTEN=?,MODUS=?,CNUMBER=?,LMASSNO=?,XREG=?,XREG2=?,BREG=?,CABLPAGE=?
122275   DOUBLE BLSTX=?
122275   INTEGER POINTER LREG
122276
122276   % ENTRY: D=NUMBER OF PAGES TO TRANSFER
122276   %         X=CORMAP ELEMENT OF FIRST PAGE TO TRANSFER
122276   %         A=READ/WRITE FLAG
122276   %         B=SEGMENT ADDRESS
122276   %
122276   XTRNSEG: A=:MODUS:=RTREF=:SRTREF:=D; D=:WRITTEN; CALL RWSEGM
122304           GO PSTUPR
122305
122305
122305   TRNSEG: A=:MODUS:=L=:"LREG"; X=:XREG
122311           D=:WRITTEN; NUMBER=:CNUMBER; X:=BPAGLINK
122315           FOR CNUMBER DO WHILE X><0
122316              T:=CORMBANK; *LDATX DPGPR
122320              IF A=:D/\160000\/MODUS=0 OR MODUS SHR 5WIP/\D><0 THEN
122330                 1; CALL RWSEGM
122332              FI; T:=CORMBANK; *LDXTX DPAGL
122334           OD
122336           IF WRITTEN><0 THEN X:=SRTREF; LMASSNO; CALL RELEASE FI
122343           WRITTEN; X:=XREG; GO LREG
122346   *)FILL
122356
122356   INTEGER WRITTEN,MODUS,CNUMBER,LMASSNO,XREG,XREG2,BREG,CABLPAGE,NPAGES,BMASS
122370   INTEGER POINTER LREG2
122371   DOUBLE BLSTX
122373
122373   RWSEGM: A=:NPAGES:=L=:"LREG2":=B=:BREG; X=:XREG2
122401           IF WRITTEN=0 THEN
122403              MADR SHZ -16; MASSUNIT(A) SH 6+MODUS+60=:SWPMTMODUS
122413              ABLPAGE(X)=:CABLPAGE*NPAGES=:SWPBLPAGE
122417              X=:T+X; DBLST(X)=:BLSTX; X:=T
122424              IF D=0 AND A=0 THEN CALL ERRFATAL FI
122430              MASSNO(X)=:LMASSNO; CALL LOGPH; A=:BMASS=:B
122435              X:=SRTREF; CALL BRESERVE
122437              IF A<0 THEN
122440                 IF SWPFLAG><0 THEN
122442                    BMASS=:B; GO LNKISWAP
122445                 FI; SCLOAD; CALL RELEASE
122447                 IF X=RTREF GO PRWAIT; GO PSTUPR; *)FILL
122474              FI; MIN WRITTEN; BREG=:B
122477           FI; T:=CORMBANK; X:=XREG2; *LDATX DALOG
122502           A=:D:=LOGADR/\377; D-A; MADR/\37777+D*CABLPAGE=:L
122513           BLSTX; D+L; A:=A+C; AD=:SWPMASSADR
122517           *LDATX DPAGP
122520           D:=0; AD SHZ -6=:SWPCOREADR
122523   *"BSWLG
122523            IF CSWLG><0 AND MODUS BIT "0" THEN          % COUNT PAGES WRITTEN BACK
```

```
122530              *MIN I (SNWP2; SKP; MIN I (SNWP1; JMP *+1
122534           FI
122534    *
122534            A:=BMASS=:B:="SWPPARLIST"; CALL PMTRANS; IF HSTAT BIT 4 GO ERR
122543           X:=BREG=:B:=XREG2; T:=CORMBANK
122547           IF MODUS><0 THEN
122551              *LDATX DPGPR
122552              A BZERO SWIP
122553           ELSE
122554              *LDATX DPGPR
122555              A=:D:=FLAG/\177400\/D
122561           FI; *STATX DPGPR
122562           GO LREG2
122563    *)FILL
122576
122576    % DISC TRANSFER ERROR
122576    % ABORT PROGRAM CURRENTLY USING THE SWAPPER
122576    %
122576    INTEGER TRTEXT:='
122576    ***** FATAL ERROR: DISC TRANSFER ERROR IN SEGMENT HANDLING. *****
122576    '
122641    ERR:    "TRTEXT"; CALL DPRINT
122643           X:=SRTREF=:RTREF; GO PRTEXT      % ABORT PROGRAM
122646
122646    RBUS
122653
122653
122653
122653    % ============================================================================
122653    % 37.16       L I N K O V E R
122653
122653    %SUBROUTINE TO LINK PAGES OVER TO REQUESTING SEGMENT
122653    % ENTRY: B=REQ.SEGMENT (SEGREF)
122653    %         X=LRU-SEGM
122653
122653    SUBR LINKOVER
122653    INTEGER CURPAGE,CNUMBER,LRUSEG,CSEGN,CALOGNO
122660    INTEGER POINTER LREG
122661    LINKOVER: A:=L=:"LREG"; X=:LRUSEG; NUMBER=:CNUMBER; B=:A; CALL CSEGS
122670           IF FLAG BIT 5FIX THEN CALL ERRFATAL FI
122674           A:=B-SEGSTART=:D:=0; T:=5SEGSIZE; *RDIV ST
122702           IF D><0 THEN CALL ERRFATAL FI; A=:CSEGN; T:=CORMBANK
122707           FOR CNUMBER DO WHILE X:=LRUSEG.BPAGLINK><0
122712              *LDATX DPGPR
122713              IF A BIT SWIP GO OUT
122715              X=:CURPAGE; *LDATX DPAGL
122717              A=:LRUSEG.BPAGLINK
122721              IF X:=BPAGLINK=0 THEN
122724                 IF SEGLINK=0 THEN BSEGLINK=:SEGLINK; SEGREF=:BSEGLINK FI
122732                 LOGADR/\377-NUMBER
122735              ELSE
122736                 *LDATX DALOG
122737              FI; A=:CALOGNO:=CURPAGE; A:=:X; *STATX DPAGL
122743              X=:BPAGLINK; CSEGN=:D
122746              IF FLAG NBIT 5DEMAND THEN CALOGNO-1 ELSE CPNUMB FI
122755              *STDTX DALOG
122756           OD
122760    OUT:    IF LRUSEG.BPAGLINK=0 THEN CALL SREMOVE FI
122764           GO LREG
122765    RBUS
```

```
122777
122777
122777    % ============================================================================
122777    % 37.17       S R E M O V E
122777    %
122777    % SUBROUTINE TO REMOVE A SEGMENT FROM SEGMENT LINK
122777    % X=SEGMENT POINTER
122777
122777    SUBR SREMOVE
122777    SREMOVE: T:="BSEGLINK"; X=:D:=BSEGLINK
123002           DO WHILE X><D; IF X<0 GO OUT; X=:T:=X.SEGLINK OD
123010           X.SEGLINK; 0=:X.SEGLINK; A=:T.SEGLINK
123014    OUT:   D=:X; EXIT
123016    RBUS
123017
123017
123017    %-============================================================================
123017    %              S R E S E R   -   S R E L E S
123017    %
123017    % SUBROUTINES TO RESERVE AND RELEASE ALL SWAPPING RESOURCES
123017    %
123017    % ENTRY:       B=MONITOR CALL WORKING FIELD
123017    %
123017    SUBR SRESER,SRELES,ZSRELES
123017    INTEGER BRG,CINDX,XREGG
123022    INTEGER POINTER CLR
123023
123023    SRESER: A:=L=:"CLR":=B=:BRG; X=:XREGG
123030           X:=RTREF; "CLFIE"=:B; CALL BRESERVE
123034           IF A<0 THEN
123035              CALL SRELES
123036              X.STATUS BONE 6WAIT=:X.STATUS
123041              BRG=:B; OLDPAGE BONE 5SWWAIT=:OLDPAGE; GO VENT
123047    OUT:   FI; 0=:CINDX
123050           DO WHILE X:=CINDX<=3
123054              IF MASSNO(X)><0 THEN
123056                 CALL LOGPH; A=:B; X:=RTREF; CALL BRESERVE
123062                 IF A<0 THEN
123063                    CALL SRELES; CALL FREXQU; CALL TOWQU; GO VENT
123067                 FI
123067              FI; MIN CINDX
123070           OD
123071           BRG=:B; X:=XREGG; GO CLR
123075    VENT:  BRG=:B; ZPREG-1=:ZPREG; GO M61RET
123103
123103    INTEGER CPROG,XINDX,BBRG,XXRG
123107    INTEGER POINTER CLLR
123110
123110    ZSRELES: X=:CPROG=:XXRG; GO FELLS
123113    SRELES: RTREF=:CPROG; X=:XXRG
123116    FELLS: A:=L=:"CLLR":=B=:BBRG
123122           0=:XINDX
123123           DO WHILE X:=XINDX<=3
123127              IF MASSNO(X)><0 THEN
123131                 CALL LOGPH; IF A.RTRES=CPROG THEN X=:B; X:=CPROG; CALL BRELEASE FI
123142              FI; MIN XINDX
123143           OD; "CLFIE"=:B; X:=CPROG; IF RTRES=X THEN CALL BRELEASE FI
123153           BBRG=:B; X:=XXRG; GO CLLR
123157    RBUS
123172
```

```
123172
123172    %-------------------------------------------------------------------------
123172    % 37.18       A T R N S E G
123172    %
123172    % SUBROUTINE ON APPLICATION LEVEL TO START
123172    % TRNSEG ON MONITOR LEVEL
123172    %
123172    % ENTRY:      A=NUMBER OF PAGES TO TRANSFER
123172    %             X=ADDRESS OF CORMAP ENTRY
123172    %             D=READ/WRITE FLAG
123172    %             T=SEGMENT ADDRESS
123172    %
123172    SUBR ATRNSEG
123172    INTEGER CT,CA,CD,CX,CB; TRIPLE CTAD=CT
123177    INTEGER POINTER CLR
123200    ATRNSEG: TAD=:CTAD; X=:CX;=L=:"CLR";=B=:CB
123206    LOOP:   *IOF
123207            "YTRNSEG"; *IRW MLEVB DP
123211            CT; *IRW MLEVB DB
123213            CX; *IRW MLEVB DX
123215            CD; *IRW MLEVB DA
123217            CA; *IRW MLEVB DD
123221            MLEV; *MST PIE; MST PID
123224            CALL STMLEV; MLEV; *MCL PIE
123227            TAD:=CTAD; X:=CB=:B:=CX
123233            GO CLR
123234    RBUS
123236
123236    %-------------------------------------------------------------------------
123236    % 37.21       O V E R L A P
123236    %
123236    SUBR OVERLAP
123236    % SUBROUTINE TO SEARCH FOR OVERLAP AND DELETE PAGES
123236    % ENTRY:      A=SEGM. ELEMENT POINTER, T=REENTRANT SEGMENT
123236    %
123236    %
123236    INTEGER SEGAB,XREG,RSTART,REND,RESEG,PREVT
123244    INTEGER POINTER SEG1:=XSGRT,BP1:=BPAG1    % SEGMENT 1
123246    INTEGER POINTER LREG
123247
123247    OVERLAP: IF A=0 THEN EXIT FI
123251            A=:SEGAB; T=:RESEG; X=:XREG:=L=:"LREG"; IF A.SEGLINK=0 GO OUT
123261            RESEG.LOGADR/\377=:RSTART; X.LOGADR SHZ -10+RSTART=:REND
123271            X:=SEGAB+1; T:=0
123274    PSTART: DO
123274               X=:D; T=:PREVT; *LDXTX DPAGL
123277            WHILE X><0
123300               T:=CORMBANK; *LDATX DALOG
123302               IF A>=RSTART AND A<REND THEN          % OVERLAP
123310                  T:=CORMBANK; CALL CLPAGE; *LDATX DPAGL
123313                  D:=:X; T=:L:=PREVT; *STATX DPAGL          % REMOVE PAGE
123317                  X:=:D; T:=:L; BP1; *STATX DPAGL     % LINK TO DUMMY SEGMENT
123323                  X=:BP1; A:=1; *STATX DPGPR          % SET SEGMENT NUMBER
123326                  IF SEG1=0 THEN             % MOVE DUMMY SEGMENT TO END OF SEGMENT LINK
123330                     BSEGLINK
123331                     DO WHILE A.SEGLINK><-1 OD
123337                     "SEG1"=:X.SEGLINK; T=:SEG1
123342                  FI; X:=D; T:=L; GO PSTART
123345               FI; T:=CORMBANK
123346            OD
```

```
123347          IF SEGAB.BPAGLINK=0 THEN CALL SREMOVE FI
123353  OUT:    X:=XREG; GO LREG
123355  RBUS
123362
123362  % ============================================================================
123362  % 36.9          S H R S O V E R L A P
123362  %
123362  % SUBROUTINE TO CHECK IF A REENTRANT SEGMENT ADDRESS OVERLAPS
123362  % A NON DEMAND SEGMENT
123362  % IF SHRSOVERLAP;
123362  % ENTRY:        X=SEGMENT TABLE ADDRESS OF ACTUAL NONREENTRANT SEGMENT
123362  %               A=SEGMENT FLAG OF ACTUAL NONREENTRANT SEGMENT
123362  %               T=REENTRANT SEGMENT NUMBER
123362  %
123362  % EXIT:         NO OVERLAP
123362  %
123362  % ERROR AND EXIT TO RETXIT IF OVERLAP
123362  %
123362
123362  SUBR SHRSOVERLAP
123362  INTEGER CSTRA,CLA,CLRS
123365
123365  SHRSOVERLAP:   IF A BIT 5DEMAND OR T=0 GO OUT
123371                 A:=T*5SEGSIZE+SEGSTART:=:X=:T
123376                 X.LOGADR/\377=:CSTRA; X.LOGADR SHZ -10=:CLA; X:=T
123405                 IF X=0 GO OUT
123406                 X.LOGADR SHZ -10=:CLRS; X.LOGADR/\377
123413                 IF A>CSTRA THEN
123416                    IF A-CLA<T GO 45ERR
123421                 ELSE
123422                    IF A+CLRS>T GO 45ERR
123425  OUT:           FI; EXIT
123426  45ERR: CALL 9ERRA(#45); GO RETXIT
123431
123431  RBUS
123436
123436
123436
```

```
23436
23436   %-------------------------------------------------------------------------
23436   %
23436   %   3 8 . 0     S Y S T E M   S T A R T   A N D   S T O P
23436   %
23436   %-------------------------------------------------------------------------
23436   % 38.1        P W F A I L   R E S T A R T   S E T P T A B L
23436   %
23436
23436   INTEGER ARRAY PCCS:=(2,12,22,32,42,3652,62,72,102,112,3722,132,3742,3752,3762,2)
23456
23456   INTEGER ARRAY LEV1=?
23456   INTEGER LEV14=?,URERTP=?
23456   INTEGER PIEREG,PIDREG
23460   SUBR PWFAIL,RESTART,SETPTABL
23460
23460   %INTEGER PDELY:=6000            % DELAY COUNTER
23460   INTEGER LEVC,SRBI(0); *SRB 10
23462   INTEGER ARRAY QMSDF:=(DRFIE,DRFI2,BIGDI,BIGD2,BIGD3,BIGD4,GIGDI,GIGD2,WIGDI,WIGD2,-1)
23475   DISP 0; REAL FSO, RCTRG=CTRG, RTRG=TRG; PSID
23475
23475   %ROUTINE ON LEVEL 14, ACTIVATED BY POWER FAIL
23475   %\AN ALSO BE CALLED BY THE STOP-SYSTEM COMMAND
23475   PWFAIL: *PIOF
23476   *"8N500
23476           CALL 5PF1
23477   *"
23477           -15=:LEVC; SRBI; X:="LEV1"
23503          FOR LEVC DO; *EXR SA
23504                X+10; A+10
23506          OD
23510          *TRA PIE
23511          A=:PIEREG; *TRA PID
23513          A=:PIDREG
23514          CALL PFXMSG
23515   *"8N500
23515           CALL 5PF2
23516   *"
23516           IF SEGMC><0 AND A.FLAG BIT 5WPM AND X:=X.BPAGLINK><0 THEN
23526             T:=CORMBANK
23527             DO
23527                 *LDBTX DALOG; 177700                % MIC. PROGRAM BUG!!!!!
23531                 AD:=DOU0; *STDTX DPGPR
23533                 *LDXTX DPAGL
23534             WHILE X><0
23535             OD
23536          FI
23536   *"8N500
23536           CALL 5PF3
23537   *"
23537    % SAVE WINDOW ENTRIES IN PAGE TABLE 3
23537          B:=-10                                    % B:=177770
23540          X:="SVPT3"
23541          DO WHILE B><0; DOU0=:X.DOU0; X+2; B+2; OD
23550          FOR X:=-2000 DO; OD                       % DELAY BEFORE STOP
23552   PWAIT: *WAIT; JMP *
23554
23554
23554   %  - - POWER UP AGAIN:
23554   RESTART:
```

```
123554   %       FOR A:=0 TO PDELY DO; FOR X:=-1000 DO; OD; OD    % DELAY AFTER POWER UP
123554           2000; *TRR IIE; TRA IIC  %ENABLE POWER FAIL
123557           *SEX
123560           IF HWINFO(0)/\377=2 THEN
123566              *143500                                  % ND-100 CX.
123567           FI
123567
123567   % IF DISK DMA WAS ACTIVE AT POWER FAIL TIME, FORCE NEW CALL
123567
123567           X:="QMSDF"; X=:B
123571   QL0:    X:=S0; IF X=-1 GO SETPTABL; IF X=0 GO QL1
123576           100=:X.DIFTCOUNT                  % WAIT 100B*3 SECS. BEFORE TIMOUT ON DISC
123600           0=:X.BUSFL              % FORCE NEW CALL
123601           A:=X.TACNS=:X.TACOU; TAD:=X.RCTRG=:X.RTRG; A:=X.CXRG=:X.XRG
123607   QL1:    B+1; GO QL0
123611   *)FILL
123631
123631   %   COMMON FOR INITIAL START AND RESTART:
123631
123631   SETPTABL: 0=:PWRFREST; 177000=:B; A:=162000; D:=0
123636   % MAP RESIDENT INTO PIT 0 AND PIT 3
123636   % AND CLEAR THE REST OF THE PAGE INDEX TABLES
123636           X:=177600
123637           DO AD=:DOU0=:X.DOU0; D+1; B+2; X+2; WHILE D<=LRESP OD
123650           DO WHILE B><177600
123653              0=:S0=:S1; B+2; *TRA IIC
123657              IF A=12 GO PWAIT
123662           OD; DO WHILE X><0; 0=:X.S0=:X.S1; X+2; OD
123670   %  *"CXCPU
123670   % MAP XMSG AND TERMINAL DRIVER INTO PIT 3
123670           A:="9SMRE" SHZ -12=:D
123673           A:="9EPT3" SHZ -12=:L; IF "9EPT3"/\1777><0 THEN L+1 FI
123702           X:=177600+D+D; A:=162000
123706           DO AD=:X.DOU0; WHILE D><L; X+2; D+1; OD
123714   %  *"
123714           GO PT3L1; *)FILL
123724   PT3L1:
123724   % SET BUFFER WINDOW PIT ENTRY TO ZERO
123724           "5BFPAGE" SH 1+177000=:B; A:=0=:D; AD=:DOU0
123733   % SET SYSTEM SEGMENT PIT ENTRIES TO ZERO
123733           "5SSSZ"-=:X; A:="SYSST" SH 1+177000=:B; A:=0=:D
123744           FOR X DO AD=:DOU0; B+2 OD; CALL CLERT
123750   % INITIALIZE PCR ON ALL LEVELS
123750           FOR X:=0 TO 17 DO
123754              PCCS(X); *TRR PCR
123756           OD
123760           IF PIDREG><0 THEN        %AFTER POWER FAIL
123762   % RESTORE PIT ENTRIES FOR RT'S SYSTEM SEGMENT.
123762              IF BACKGROUND=0 THEN CALL SETRT FI
123765              SEGMA; CALL RSPTABL; SEGMB; CALL RSPTABL; SEGMC; CALL RSPTABL
123773   +"BXMSG
123773    % RESTORE PIT ENTRIES FOR SEGMENT #33
123773              IF "SG33".FLAG BIT 5FIX THEN
123777                 X:=X.BPAGLINK; T:=CORMBANK
124001                 DO WHILE X><0
124002                    *LDATX DALOG
124003                    A/\77 SH 1+177600=:B
124007                    *LDDTX DPGPR
124010                    A BONE 5WIP; AD=:DOU0
124012                    *LDXTX DPAGL
```

```
124013                OD
124014             FI
124014    *"
124014    % RESTORE PIT ENTRIES FOR PIT3-SEGMENT
124014             X:="SG41".BPAGLINK; T:=CORMBANK
124017             DO WHILE X><0
124020                *LDATX DALOG
124021                A/\77 SH 1+177600=:B
124025                *LDDTX DPGPR
124026                AD=:DOUO
124027                *LDXTX DPAGL
124030             OD
124031
124031    % RESTORE WINDOW ENTRIES IN PAGE TABLE 3
124031             B:=-10                                  % B:=177770
124032             X:="SVPT3"
124033             DO WHILE B><0; X.DOUO=:DOUO; B+2; X+2; OD
124042             CURPROG.ACTPRI /\3773; *TRR PCR
124046             "ERR14"=:LEV14
124050          FI
124050          CALL TBUSPEED
124051          GO CLADV
124052    *)FILL
124100    INTEGER ARRAY SVPT3(10)       % PIT 3 WINDOW ENTRIES (PAGE 374 - 377)
124110    INTEGER 5IX:=0, 500IOX
124112    INTEGER ARRAY 5IOXARR(0)      % ND-500 IOX TABLE
124112    *"8N500; 60  ;"               % ASCENDING IOX PLEASE !
 124113     *"8CP55; 560 ;"
 124113    *"8CP53; 660 ;"
 124113    *"8CP54; 760 ;"
 124113    *"8CP52; 1060;"
 124113    *"
 124113    * -1
124114
124114    % CLEAR ALL DEVICES EXCEPT CUSTOMER RESERVED DEVICES
124114
124114    INTEGER CCIOX
124115    CLADV: 5IOXARR(0)=:500IOX
124120           0=:CCIOX
124121           DO
124121              IF CCIOX=500IOX THEN                 % ND-500 DEVICE NUMBER
124125                 CALL 5MCST; X:=5IX+1=:5IX; 5IOXARR(X)=:500IOX
124133                 CCIOX+20=:CCIOX; GO NXIOX
124137              FI; X:=0
124140              DO WHILE X<<"USDVSIZE*2"             % DO NOT CLEAR USER RESERVED DEV.NO'S
124143                 AD:=USIOXTAB(X)
124144                 IF A>>=CCIOX AND D<<=T THEN
124151                    A:=D/\177774=:CCIOX; GO NXIOX
124155                 FI; X+2
124156              OD; T:=CCIOX+"DCONT"; A:=20; *IOXT    % CLEAR DEVICE
124163    NXIOX: WHILE CCIOX+4><0
124166              A=:CCIOX
124167           OD; GO L1
124171    *)FILL
124175    INTEGER CCNTRG
124176
124176    L1:
124176           LGCOLDSTART; CALL LOGPH
124200           IF A=0 OR A.TYPRING NBIT 5TERM THEN X:="DT01R" FI
124206           T:="CNTREG"; CALL XGTDFADDR; A BZERO "0"=:CCNTRG
```

```
124212          T:="HDEV"; CALL XGTDFADDR; T:=A+DCONT; CCNTRG; *IOXT
124220          IF VDDFLAG = 0 THEN
124222
124222    %        ONLY IF SWAPPING NOT DONE VIA VDD/VDP PAIR.
124222    %        CLEAR SWAPPING DEVICE (CLEAR DEVICE AND READ STATUS, THEN WAIT FOR READY)
124222
124222             MASSNO(0); CALL LOGPH; A.HDEV+"164000+5"=:T; 20; *EXR ST
124233             IF T BIT 11 THEN A:=34004; *EXR ST     % RETURN TO ZERO SEEK IF BIG DISC
124237             FI; T-1                                % READ STATUS (4=READ STATUS)
124240    DDLAY:   FOR X:=-200 DO; OD                     % DELAY
124242             *EXR ST; BSKP ZRO 160 DA; JMP ADLAY    % ON CYLINDER
124245             *BSKP ONE 030 DA; JMP DDLAY            % UNTIL ON CYLINDER
124247          FI
124247    ADLAY: *BSET ZRO SSZ
124250          *TRA IIC; TRA PEA; TRA PGS
124253          "0"; *TRR PID; TRR PIE
124256          "LVO"; *IRW 0 DP; ION
124261    LVO:  *IOF
124262          IMASK; *TRR IIE
124264    % SET CACHE INHIBIT AND CLEAR:
124264          IF CPSTA BIT 5N100 THEN
124267            LCACHLIM; *TRR 11              % LOWER LIMIT
124271            UCACHLIM; *TRR 12              % UPPER LIMIT
124273          ELSE
124274            CACHLIM; *TRR 12
124276          FI; *TRR 10
124277          GO BYPAX; *)FILL
124320
124320    %%%   INITIALIZE BUS EXPANDER
124320
124320    INTEGER CURBX                          % TRACKING OF BUS EXPANDER ADDRESS
124321
124321    BYPAX:
124321    *"8BEX1
"124321    %  INITIALIZE MULTIPORT IV
124321
124321          IF CPSTA BIT 5N100 THEN                  % IF ND100
124324             T:= 100200 =: CURBX                   % INIT FIRST DEVNO
124326             DO WHILE 100400 >> CURBX              % CLEAR AND INIT
124332                T+4 =: CURBX - 1                   % ALL POSSIBLE MPM4 BUSC'S
124335                A := 040; * IOXT                   % ENABLE READ LPEA
124337                T - 3; * IOXT                      % READ LPEA
124341                T + 2; * IOXT                      % READ STATUS TO CLEAR IT
124343                T + 1; A := 06; * IOXT             % CLEAR DEVICE AND ENABLE INTERRUPT
124346             OD
124347
124347             * TRA IIC                            % CLEAR IIC
124350
124350          FI
124350    *"
"124350
124350          GO L2; *)FILL
124354
124354    % RESTORE REGISTER BLOCKS
124354    INTEGER LEVCN,LRBI(0); *LRB 10
124356    L2:   X:="LEV1"; -16=:LEVCN; LRBI
124362          FOR LEVCN DO; *EXR SA
124363                X+10; A+10 OD
124367          "FREQU+160001"; *IOX DCONT RTCLD
124371          9TIM0; *IOX DDW RTCLD
```

```
124373          PIEREG; *TRR PIE
124375          PIDREG ; *TRR PID
124377          IF PIDREG><0 THEN
124401    *"8CLI1+8CLI2+8CLI3+8CLI4+8CLI5+8CLI6
 124401                  IF X:=URESTART><0 THEN; *JPL ,X
124404                  FI
124404                  IF URPRO><0 THEN A=:URERTP
124407                    "URERTP"+3=:B; MQUEU=:MLINK; A:=B=:MQUEU
124416                  FI
124416          FI
124416    @LIB CXCPU
124416          162000=:"177600+L12LGP+L12LGP".SO          % WINDOW FOR LEVEL 12
124421          162000=:"177600+L10LGP+L10LGP".SO          % WINDOW FOR LEVEL 10
124424    @LIB
124424          GO PENTO
124425    *)FILL
124440
124440    % SUBROUTINE TO SET SPEED FOR TERMINAL BUFFER AND MULTITERMINAL INTERFACE
124440    % THE SPEED IS FOUND IN INPUT DATAFIELD; -1 MEANS DO NOT SET SPEED
124440
124440    INTEGER POINTER INDXX,CLREG; INTEGER ICNTI,CHDEV
124444    DISP 12; INTEGER CCDFILD; PSID
124444    TBUSPEED: A:=L=:"CLREG"
124446          X=:D; "TMRTE"-2=:"INDXX"
124452          DO WHILE X:=INDXX><-1 OR "INDXX"="TMRTE"  % IN TIMER TABLE
124462                IF X><0 AND X><-1 THEN
124466                   IF X.TYPRING BIT 5TERM THEN       % TERMINAL
124471                      T:="HDEV"; CALL XGTDFADDR; A=:CHDEV
124474                      IF A NBIT 2 THEN
124476                         T:="TSPEED"; CALL XGTDFADDR
124500                         IF A+1><0 THEN
124502                            T:=CHDEV+1; A-1; *IOXT
124506                FI FI FI FI; MIN "INDXX"
124507          OD
124510          1=:ICNTI
124512          FOR ICNTI TO MXLIN DO          % SET SPEED FOR SIN/SIN COMMUNICATION
124516            X:=LINAR(ICNTI); IF X.TSPEED><-1 THEN
124524              T:=X.CCDFILD.HDEV+1; *EXR ST
124530            FI
124530          OD D=:X; GO CLREG
124536
124536    %SUBROUTINE TO SET CONTENTS TO PAGETABLE
124536    INTEGER XWIP
124537    RSPTABL: IF =0 THEN EXIT FI; A=:D
124542          IF A=SEGMC AND A.FLAG BIT 5WPM AND X:=X.BPAGLINK><0 THEN
124553             T:=CORMBANK
124554             DO
124554                *LDBTX DALOG; 177700; LDDTX DPGPR         % MIC. PROGRAM BUG!!!!
124567                AD=:DOUO; *LDXTX DPAGL
124561             WHILE X><0
124562             OD
124563          ELSE
124564             IF D.FLAG BIT 5WPM THEN 10000 ELSE A:=0 FI; A=:XWIP
124574             IF X:=X.BPAGLINK=0 THEN EXIT FI; T:=CORMBANK
124601             DO
124601                *LDBTX DALOG; 177700; LDDTX DPGPR         % MIC. PROGRAM BUG!!!!
124604                A\/XWIP; AD=:DOUO; *LDXTX DPAGL
124607             WHILE X><0
124610             OD
124611          FI; EXIT
```

```
124612    RBUS
124622    @ICR
124622    INTEGER ARRAY LEV1:=(
124622    ERRFA,0,0,0,0,0,0,0,
124632    ERRFA,0,0,0,0,0,0,0,
124642    MONEN,0,0,0,0,0,0,0,
124652    ERRFA,0,0,0,0,0,0,0,
124662    ERRFA,0,0,0,0,0,0,0,
124672    ERRFA,0,0,0,0,0,0,0,
124702    ERRFA,0,0,0,0,0,0,0,
124712    ERRFA,0,0,0,0,0,0,0,
124722    ERRFA,0,0,0,0,0,0,0,
124732    ID10+5,0,0,0,0,0,0,0,
124742    ID11+5,0,0,0,0,0,0,0,
124752    ID12+5,0,0,0,0,0,0,0,
124762    ID13+5,0,0,0,0,0,0,0);
124772    INTEGER LEV14:=(
124772    ENT14,0,0,0,0,0,0,0);
125002    @CR;
125002
125002
125002
125002    DOUBLE ARRAY XCCTAB(0); * *+CCNO+CCNO+20/
125024
125024
125024    *"BLAMU
```

```
125024    %
125024    %-------------------------------------------------------------------------
125024    %       L A M A R R
125024    %
125024    % PHYSICAL MEMORY PARTS AVAILABLE FOR LAMUS
125024    %
125024    % FORMAT: 2 LOCATIONS FOR EATCH ENTRY
125024    %          FIRST PHYSICAL PAGE IN AREA
125024    %          LAST PHYSICAL PAGE IN AREA
125024    %
125024    INTEGER ARRAY LAMARR(NINSZ*2)
125074    * LAMAR<*; )ZERO
125074    *
 125074
125074
125074    %-------------------------------------------------------------------------
125074    % 36.24       N S W P A G E  -  N I N I T P A G E  -  C U M T A B L E
125074    %
125074    % NSWPAGE     : MEMORY AREA NOT USED AS SWAPPING AREA
125074    % NINITPAGE   : MEMORY AREA NOT LEGAL TO INITIALIZE
125074    %
125074    % THE LAYOUT IN THE TABLES:
125074    %       TWO WORDS FOR EACH ENTRY, FIRST PAGE AND LAST PAGE IN AREA
125074    %       THE TABLES ARE TERMINATED BY A ZERO
125074    %
125074    DOUBLE ARRAY NSWPAGE(NNSWSZ)            % MEMORY AREAS UNAVAILABLE FOR SWAPPING
125144    *NINSZ
125145    DOUBLE ARRAY NINITPAGE(NINSZ)           % MEMORY AREAS NOT ACCESSED IN START-UP (SHARED BETWEEN CPU'S)
125215    DOUBLE ARRAY CUMTABLE(CUMSIZE)          % CUSTOMER RESERVED MEMORY AREAS
125241    DOUBLE ARRAY USIOXTAB(USDVSIZE)         % CUSTOMER RESERVED PHYSICAL DEVICE NUMBERS (SHOULD NOT BE CLEARED)
125261    * NSWPA<*; )ZERO
125261    * 8VDD+8VDP
125261
125261    @DEV 1
125261    @DEV (S-S-J)ND500-DRIVER-1
```

```
125261    %
125261    *'8N500
'125261    *)KILL YENDC; YENDC=*
125261    */ENDC/
066364
066364    %MAC

)9SCLL
066364
066364    %       MACRO FOR ND-500 CPU SPECIFIC DATA FIELD                    %M5
066364
066364    )M.DEF M5CPU $NO,$500DV

)KILL E5CPU
ZXC=*; N500D-24/$NO ;ZXC/
)KILL ZXC
        $NO ;0;N500D-20;-1
        0;0,N500T;0;-3;$500DV ;N500;N500
E5CPU=*
5CPU$NO , 0;0;*-2;2;0;0;0;0
        0;1%$NO ;200;0;-1;0;0;0
        0;-1;-1;-10;0;0;0;0
        0
)
066364
066364
066364
066364    %       N 5 0 0   D A T A F I E L D                                 %M5
066364
066364            -1;-1;0
066367            0;0;-1;-1;0;0;0;0;
066377            0;0;0;0;0;21;0;0
066407            0;0;0;S500S;0;0;62;0
066417    N500D.  0;0;*-2;2;0;0;0;0
066427            0;0;0;0;0;0;0;0
066437            0;0;0;0;0;0;0;0
066447            0;0;0;0;0;0;0;0
066457            0;0;0;0;0;0;0;0
066467            0;0;0;0;0;0;0;0
066477            -1;0;;0;0;0;0;0;0
066507            0;0;0;0;0;0;0;0
066517            0;0;0;0;0;0;0;0
066527            0;1000;100;104;0;0;0;0
066537            0;0;0;0
066543            5GNSE;RESNA;RELNA;ROBJE;WOBJE
066550            PUT1L;SETIO;ACTDR;DIRTA;NAMTA;5GETL;LOGPH;5ATRA
066560            MERRC;XONES;XOFFE;CUSER;TTNO;GMAIN;GUSEN;MDEAB
066570            WORKA;FSYSI;SEGMO;TTIFI;BCHFL;PASST;GFILI;GFIAC
066600            DIRUN;CCTAB;PRJN;GPUPI;CMXAC;GMUSI;5GSYD;GETLS
066610            RSGMO;MMREE;CPNT;TS3CO;CLADB;5DFRE;5DLRE;CT500
066620            XLOCK;XUNLO;5S1TS;5R1TS;S5TSL;WBACK;5MBBA;RTPWO
066630            5DTUP;0;0                                                   % 2 SPARE LOCATIONS
066633
066633    S5CPU=*+14
066633    E5CPU=*
066633    '8CP51; M5CPU 1,60
066633
066633    )KILL E5CPU
066633    ZXC=*; N500D-24/1;ZXC/
066633    )KILL ZXC
```

```
066633           1;0;N500D-20;-1
066637           0;0;N500T;0;-3;60;N500;N500
066647  ESLPUE=*
066647  SCPU1,  0;0;*-2;2;0;0;0;0
066657           0;1@1;200;0;-1;0;0;0
066667           0;-1;-1;-10;0;0;0;0
066677           0
066700
066700   )8N500
066700   )9RCLC
)9SLPL066700    *)KILL 7ENDC; 7ENDC=*
066700   *YENDC/
```

```
125261
125261    INTEGER SGBRKADDR=?,DBASE=?,KEXMC=?,5LREG=?,5AREG=?
125261    INTEGER ARRAY FUNCS=?,5IFUNC=?,ADRPROC=?,N500REENT=?
125261    INTEGER CMXACTPROC=?,5HRTP=?,5HRTB=?,5MLOG=?,5MLOPROC=?
125261    INTEGER 5HIFLAG=?,5HINTERVAL=?,5HICHANNELS=?,5LOGPROC=?
125261    INTEGER TS5STOP=?,SWMSG=?,5MSINIT=?,PROCADD=?
125261    INTEGER C5REENT=?,CTTIFIELD=?,CERNENABLE=?,5DBFLAG=?
125261    DOUBLE 5HISTART=?,5HIOUTSIDE=?
125261    DOUBLE ARRAY 5HIDATA=?
125261
125261    INTEGER 5TTIFIELD=?,5PASSTYPE=?,5BCHFLAG=?,5SPASSTYPE=?
125261    INTEGER ARRAY 5TSLNTIME=?,5TSLSTATUS=?,5TSLCOUNTA=?
125261    INTEGER HIMESS=?,5CSTCK=?
125261
125261    @LIB CXCPU
125261    @ICR;
125261    SUBR N500M,5ATRANS,5ALTON,5GMAIN,5GUSEN,SEGMONC,5ATSIBMOVE,5STAMOVE,
125261           N500C,N500SCHEDULER,N500TMR,N500INCR,5BABORT,ESC500,
125261           5IORESTART,5ESCRESTART,ESC5ON,FROMESC,5GNSEG,5GETL,500HIST,
125261           CH5MXTIME,RESNAMSEG,RELNAMSEG,5PF1,5PF2,5PF3,5SWRT,5MCST,5GSYD,
125261           XOFFESCAPE,XONESCAPE,RSTARTALL,TER500,MONICO,5PUTL,GETLS,
125261           5BPUT,IBMOVE,NW5ST,RSGMONC,FSEMA,MMBREACTIVATE,APLRSTART,T5RST,TS3COM,
125261           PT5RST,PNW5ST,S5TSLTYPE,5S1TSLICE,5R1TSLICE,CH5CPUPRESENT,RL5PDESC,
125261           1RL5PDESC,RIOWA,5DTUP;
125261    @ICR;
125261
125261    % ************************************************************************
125261
125261    %       MAILBOX DESCRIPTION:
125261
125261    % ************************************************************************
125261
125261    % DISPLACEMENTS USED IN THE LDATX,STATX... INSTRUCTIONS
125261    %
125261    SYMBOL XLNK1=00                % LINK1
125261    SYMBOL XLNK2=10                % LINK2
125261    SYMBOL XN5ST=20                % N5STATUS
125261    SYMBOL X5SND=30                % SENDER
125261    SYMBOL X5REC=40                % 5RECEIVER
125261    SYMBOL X5SIZ=50                % 5SIZE
125261    SYMBOL XMICF=60                % MICFUNC
125261    SYMBOL XSWFU=70                % SWFUNC
125261
125261
125261    %       STANDARD PART FOR ALL COMMANDS:
125261
125261    DISP 0
125261           INTEGER                 LINK1,LINK2          % N100 PHYSICAL ADDRESS
125261                 DOUBLE            LINK=LINK1           % LINKING ALL MESSAGES
125261    PSID
125261    DISP 3
125261           INTEGER                 SENDER               % N100 PROCESS NO.
125261           INTEGER                 5RECEIVER            % N500 PROCESS NO.
125261           INTEGER                 5SIZE                % BYTES IN DATA AREA
125261           INTEGER ARRAY           XDATA(0)
125261           INTEGER                 MICFUNC              % COMMAND TO N500 MICRO-PR
125261    PSID
125261
125261    %       WRITABLE CONTROL STORE FUNCTIONS:
125261
```

```
125261   DISP 7
125261           INTEGER                WCSAD                % WCS ADDRESS
125261           INTEGER ARRAY          CSBUFFER(11)           % (11 * 16)BIT TO/ FROM WCS
125261   PSID
125261
125261   DISP 7
125261           INTEGER                N5CNT                % 5015 (N-500) STATUS/CNTREG
125261           INTEGER                N1STA                % 3022 (N-100) STATUS REG.
125261   PSID
125261
125261   %%%  FOR ND-500 MEMORY READ/WRITE
125261   %%
125261   %%       DISP 7
125261   %%             DOUBLE N500ADR              % ND-500 ADR IN READ/WRITE MESSAGES
125261   %%             DOUBLE N100ADR              % ND-100 ADR IN READ/WRITE MESSAGES
125261   %%             INTEGER NOBYTES             % NO. OF BYTES IN READ/WRITE MESSAGES
125261   %%       PSID
125261
125261
125261           SYMBOL 5MXPROC=        76          % MAX NO. OF PROCESSES  = 62D
125261
125261   %       STATUS CODES:
125261   %
125261   %             FLAG BITS FOR POWER FAIL
125261
125261           SYMBOL POWDET=         15          % POWER OFF DETECTED BY OTHER PART
125261           SYMBOL POWUP=          16          % POWER UPP MESSAGE TO ALL USERS
125261           SYMBOL POWDOWN=        17          % POWER DOWN MESSAGE TO ALL USERS
125261           SYMBOL KPOWDOWN=       100000
125261
125261           SYMBOL MFREE=         0           % MAIL BOX FREE
125261   %       SYMBOL MSGN500=        1           % MESSAGE TO NORD-500
125261           SYMBOL WAITING=        2           % N500-PROCESS IS WAITING
125261           SYMBOL ANSWER=         3           % ANSWER FROM NORD-500
125261           SYMBOL ERRANSWER=      4           % PAGE FAULT IN MESSAGE
125261           SYMBOL SWPWAIT=        5           % PROCESS IS WAITING FOR SWAPPER
125261           SYMBOL SWPPING=        6           % PROCESS IS SWAPPED
125261           SYMBOL PSWWAIT=        7           % SWAPPER PROCESS IS WATING FOR REQUEST
125261           SYMBOL MPBRKWAIT=      10          % WAITING FOR MICRO-PROGRAM BREAK
125261           SYMBOL SUSPSTAT=       11          % SUSPENDED PROCESS
125261           SYMBOL EALLOPAGE=      12          % ALLOCATE PAGE IN WSEG
125261           SYMBOL STOPPED=        13          % THE PROCESS IS STOPPED
125261                                              % EITHER BY MON 501 OR MON 502
125261           SYMBOL N5IOWAIT=       14          % THE PROCESS IS WAITING FOR OUTSTRING
125261                                              % TO TERMINAL TO BE FINISHED
125261           SYMBOL PSW1WAIT=       15          % SWAPPER PROCESS WAITING FOR ND-100 PROCESS
125261           SYMBOL I5TMQU=         16          % PROCESS IS IN ND-500 TIME-QUEUE
125261           SYMBOL WFUDMA=         17          % PROCESS IS WATING TO BE RESTARTED AFTER FAST MON UDMA
125261           SYMBOL ENAUTHORISED=   25          % YOU ARE NOT AUTHORIZED TO DO THIS
125261           SYMBOL EILPAR=         174         % ILLEGAL PARAMETER
125261           SYMBOL E301=           301         % ILLEGAL FUNCTION IN MON 61
125261           SYMBOL E302=           302         % RTCOMMON SPECIFIED IN DOMAIN,
125261                                              % BUT RTCOMMON DOES NOT EXIST IN SYSTEM
125261           SYMBOL E303=           303         % RTCOMMON'S PHYSICAL ADDRESS DOES NOT
125261                                              % MATCH THE PHYSICAL ADDRESS OF THE DOMAIN
125261           SYMBOL E304=           304         % ERROR IN LINKING TO RTCOMMON
125261           SYMBOL E305=           305         % XX FIXED SEGMENT HAS NO PAGES IN MEMORY
125261           SYMBOL E306B=          306         % XX SEGMENT FIXED BUT NOT CONTIGUOUSLY
125261           SYMBOL E307B=          307         % XX SEGMENT FIXED IN WRONG PHYS. ADDRESS
125261           SYMBOL E310=           310         % MEMORY AREA NOT AVAILABLE FOR N500 SEGMENT
```

```
125261          SYMBOL E311=            311     % TRYING TO LINK TO A NON-EXISTING
125261                                          % SINTRAN III SEGMENT
125261          SYMBOL E312=            312     % RTCOMMON SIZE SPECIFIED DOES NOT
125261                                          % MATCH THE ACTUAL RTCOMMON SIZE
125261          SYMBOL E313=            313     % XX TRYING TO LINK TO A DEMAND SEGMENT
125261                                          % IN SINTRAN III
125261          SYMBOL E314=            314     % RTCOMMON NOT CONTIGUOUS
125261          SYMBOL E315=            315     % XX SHARED SEGMENT DOES NOT OVERLAP
125261                                          % N500 SEGMENT
125261          SYMBOL N5TIMOUT=        2000    % N500 TIME-OUT
125261          SYMBOL ILMICFUNC=       2001    % ILLEGAL MICRO FUNCTION
125261          SYMBOL ILN5STATUS=      2002    % ILLEGAL STATUS IN MESSAGE TO N-500
125261          SYMBOL N5DMAERROR=      2003    % NORD-500 DMA ERROR
125261          SYMBOL ILSTOP=          2004    % ILLEGAL STOP REASON
125261          SYMBOL ILTRAP=          2005    % UNKNOWN TRAP
125261          SYMBOL SPCTRAP=         2006    % MOR/PF/PN/HE/ME/CP/MSR TRAP
125261          SYMBOL EILREG=          2007    % ILLEGAL REGISTER NUMBER
125261
125261          SYMBOL EILFUNC=         2011    % ILLEGAL FUNCTION CODE IN MON 60
125261          SYMBOL EILSEG=          2012    % ILLEGAL SEGMENT NUMBER IN LOAD
125261          SYMBOL EILFILN=         2013    % ILLEGAL FILE NUMBER IN LOAD
125261          SYMBOL EFATAL=          2014    % FATAL ERROR FROM PIT-0
125261          SYMBOL ESPRES=          2015    % N500 RESERVED FOR SPECIAL USE
125261          SYMBOL ENOPROC=         2016    % NO N500 PROCESS AVAILABLE
125261          SYMBOL ENODBUF=         2017    % NO BUFFER AVAILABLE FOR DATA TRANSFER
125261          SYMBOL EBIGBUF=         2020    % TOO BIG BYTE COUNT IN DATA TRANSFER
125261          SYMBOL ETMSHARED=       2021    % TOO MANY SHARED AREA
125261          SYMBOL ENORTC=          2022    % NO RT-COMMON DEFINED
125261          SYMBOL ESGFNCONT=       2023    % SHARED SEGMENT FIXED, BUT NOT CONTIGUOUSLY
125261          SYMBOL ESGWRADD=        2024    % SHARED SEGMENT FIXED IN WRONG ADDRESS
125261          SYMBOL ESHOUTSIDE=      2025    % SHARED AREA OUTSIDE N500 MEMORY
125261          SYMBOL EPSGBIG=         2026    % TOO BIG PROGRAM SEGMENT
125261          SYMBOL EDSGBIG=         2027    % TOO BIG DATA SEGMENT
125261          SYMBOL ENOPCOMMU=       2030    % NO PROCESS TO COMMUNICATE WITH
125261          SYMBOL ENOMEMORY=       2031    % NO MEMORY AVAILABLE FOR NORD 500
125261          SYMBOL ECSLOAD=         2032    % CONTROL STORE MUST BE LOADED
125261          SYMBOL EMDFCOM=         2033    % DEFINE-MEMORY-CONFIG. COMMAND IS REQUIRED
125261          SYMBOL EUSRON=          2034    % OTHER USER(S) ALREADY LOGGED ON N500
125261          SYMBOL ENSPRES=         2035    % N500 NOT RESERVED FOR SPECIAL USE
125261
125261
125261          SYMBOL SWADEF=          2040    % SWAP FILE ALREADY DEFINED
125261          SYMBOL NOCMASFILE=      2041    % SWAP FILE IS NOT CONTIGUOUS MASS STORAGE FILE
125261          SYMBOL SWFINUSE=        2042    % SWAP FILE IS IN USE
125261          SYMBOL SWFNFOUND=       2043    % SWAP FILE NOT FOUND
125261
125261          SYMBOL NSWFENTRY=       2045    % NO FREE SWAP FILE ENTRY
125261          SYMBOL NOMASSFILE=      2046    % NOT MASS STORAGE FILE
125261          SYMBOL SWPFATAL=        2047    % FATAL ERROR FROM SWAPPER
125261          SYMBOL MEMNAVAILABLE=   2050    % MEMORY NOT AVAILABLE
125261          SYMBOL MICFATAL=        2051    % FATAL ERROR FROM MICRO-PROGRAM
125261          SYMBOL EMONNINIT=       2052    % N500 MONITOR NOT INITIALIZED
125261          SYMBOL EZADNAVA=        2053    % N500 PAGE ZERO NOT AVAILABLE
125261          SYMBOL EIMDCONF=        2054    % ERROR IN MEMORY CONFIGURATION
125261          SYMBOL EHIUSED=         2055    % HISTOGRAM ALREADY IN USE
125261          SYMBOL EHNRESERVED=     2056    % HISTOGRAM NOT RESERVED BY YOU
125261          SYMBOL POWER=           2057    % POWER FAIL DETECTED IN NORD-500
125261          SYMBOL N5IERROR=        2060    % NORD-500 INTERFACE ERROR
125261          SYMBOL N5STOPPED=       2061    % NORD-500 STOPPED
125261          SYMBOL EPFDETECT=       2062    % POWER FAIL DETECTED IN LOADING CS.
```

```
125261          SYMBOL PFECSLOAD=       2063    % LOAD CS. AFTER POWER FAIL
125261          SYMBOL EPOWUP=          2064    % POWER UP
125261          SYMBOL EILSTYPE=        2065    % ILLEGAL LOGICAL SEGMENT TYPE IN PLACE
125261          SYMBOL ESWLOAD=         2066    % SWAPPER MUST BE PLACED
125261          SYMBOL EILPHSNO=        2067    % ILLEGAL PHYSICAL SEGMENT NUMBER
125261          SYMBOL EPFINSWP=        2070    % PAGE FAULT IN SWAPPER
125261          SYMBOL ESPTIMOUT=       2071    % TIMOUT, INPOSSIBLE TO TERMINATE N500
125261          SYMBOL EMPBREAK=        2072    % MICRO-PROGRAM BREAK REACHED
125261          SYMBOL ELOGNRESERVED=   2073    % LOGGING FACILITY NOT RESERVED BY YOU
125261          SYMBOL ELOGINUSE=       2074    % LOGGING FACILITY ALREADY RESERVED
125261          SYMBOL EN5BUFFR=        2075    % NO MEMORY AVAILABLE FOR N500 BUFFERS
125261
125261
125261          SYMBOL ENRACCESS=       2100    % NOT REQUIRED ACCESS TO THE SEGMENT
125261          SYMBOL ENIMPLEMENT=     2101    % FUNCTION NOT IMPLEMENTED
125261          SYMBOL ENAUSED=         2102    % NAME ALREADY USED
125261          SYMBOL EILOCS=          2103    % ERROR IN LOADING CONTROL STORE
125261          SYMBOL ETMFIXED=        2104    % TOO MANY FIXED MEMORY PARTS
125261          SYMBOL EDESWAP=         2105    % MASS STORAGE TRANSFER ERROR IN SWAPPING
125261          SYMBOL ETMS3FIXED=      2106    % TOO MANY SHARED SEGMENTS TO FIX
125261          SYMBOL EFUNRTP=         2136    % FUNCTION NOT ALLOWED FROM RT-PROGR.
125261          SYMBOL EIRESPL=         2140    % ILLEGAL IN RESIDENT PLACE
125261          SYMBOL ESCSTOP=         2142    % ESCAPE/BREAK TYPED
125261          SYMBOL ESNLOAD=         2143    % SYSTEM MONITOR NOT LOADED.
125261          SYMBOL E5DEBUG=         2144    % STOP-SWAPPER IN "DEBUG-MODE" (MON 377)
125261
125261  %       ERROR NUMBERS FOR ERRORS THAT PROGRAMS MAY GET
125261          SYMBOL ENOPART=         1050    % NO SWAP-FILE PART AVAILABLE
125261          SYMBOL ENOSWSPACE=      1051    % SWAPPING SPACE NOT AVAILABLE
125261          SYMBOL NOPHSEG=         1052    % NO FREE PHYSICAL SEGMENT
125261          SYMBOL EROSEGMENT=      1053    % SEGMENT NOT MODIFYABLE
125261          SYMBOL EILPROC=         1054    % ILLEGAL PROCESS NUMBER
125261          SYMBOL SWDERR=          1055    % SWAP DEVICE ERROR
125261          SYMBOL EPRMC=           1056    % PRIVILGED MONITOR CALL
125261          SYMBOL EILLSNO=         1057    % ILLEGAL LOGICAL SEGMENT NUMBER
125261          SYMBOL EPRNFOUND=       1060    % NO SUCH PROCESS
125261          SYMBOL EILADDR=         1061    % ILLEGAL ADDRESS
125261
125261  %       STOP ANSWER FROM N500
125261
125261  DISP 7
125261          INTEGER                 H500AD,L500AD       % N500 PC
125261                DOUBLE            N500AD=H500AD
125261          INTEGER                 SWFUNC=H500AD       % SPEC MESSAGE TO SWAPPER
125261          INTEGER                 CACHINFO=H500AD     % CACHE INFORMATION
125261          INTEGER                 STOPREASON          % STOP REASON
125261                DOUBLE            N100ADR=STOPREASON
125261          INTEGER                 NUMPAR              % MASK OF PARAMETERS
125261          INTEGER                 FUNCA,FUNCD
125261                INTEGER           NRBYT=FUNCA         % NUMBER OF BYTES TO READ
125261          INTEGER                 ACPROC=STOPREASON   % CURRENT PROCESS
125261          INTEGER                 WANTPRO=FUNCA       % SAMPLING PROCESS
125261                DOUBLE            FUNCVALUE=FUNCA     % FUNCTION VALUE IN MCALL
125261          INTEGER                 MCNO=FUNCA          % MONITOR CALL NO
125261          INTEGER                 KFLIP=STOPREASON    % K FLIP-FLOP
125261  PSID
125261  DISP 34
125261          DOUBLE                  SWPQ                % SWAPPING QUEUE ELEMENT
125261          INTEGER                 SCPU                % SERVING CPU
125261          INTEGER                 SMCNO               % SAVED MONITOR CALL NUMBER
```

```
125261    PSID                                                  % USED IN MON 511 SINCE MCNO IS
125261                                                          % DESTROYED DURING READ DATA MEMORY.
125261    DISP 40
125261           DOUBLE ARRAY           MONAD(20)
125261           INTEGER                HSWPFU,SWPFUNC        % SPECIAL SWAPPER FUNCTION
125261           INTEGER                HSWPST,SWPSTAT        % COM TO SWAP/STAT FROM SWAP
125261                 INTEGER SUSPPROC=SWPSTAT               % PROCESS TO SUSPEND
125261           INTEGER                HSWPINF,SWPINFO       % ADDRESS OF SWAPPING INFO
125261                 INTEGER SDUNIT=SWPINFO                 % DISK UNIT FOR SWAPP FILE
125261           INTEGER                HABFU,ABSFUFO         % ABSTRANS FUNCTION
125261           DOUBLE                 SWMEMA                % ABSTRANS MEMORY ADDRESS(N100)
125261           INTEGER                HABLC,ABLOC           % ABSTRANS BLOC NO
125261           INTEGER                HABNO,ABLNO           % ABSTRANS NO OF BLOCKS
125261           DOUBLE ARRAY           MONARRAY=HSWPFU       % MONCALL PARAMETER ARRAY
125261                 INTEGER          NPROCNO=HSWPFU        % PROCESS NUMBER
125261                 INTEGER          NMAGNO=SWPFUNC        % "MAGIC" PROCESS NUMBER
125261                 DOUBLE           DOUTDEVNO=HSWPFU
125261                 INTEGER          OUTDEVNO=SWPFUNC      % OUTPUT DEVICE NUMBER
125261                 DOUBLE           DNOBYT=HSWPST
125261    %            INTEGER          NOBYT=SWPSTAT         % NUMBER OF BYTES IN OUTSTR
125261    PSID
125261    DISP 144; INTEGER XADPROC; PSID                       % ADDRESS OF PROCESS DESCRIPTION
125261
125261    %       DISPLACEMENTS FOR THE ND-500 MICRO PROGRAM FUNCTION 26
125261    %       WRITE BYTES BACK AND START PROGRAM.
125261    DISP 15
125261           DOUBLE                 26ADDRESS             % ADDRESS IN ND-500 DATA MEMORY
125261           INTEGER                26NRBYT               % NUMBER OF BYTES
125261           DOUBLE                 26PHADR=ABUFADR       % ND-100 PH. ADDRESS
125261    PSID
125261
125261    %       DISPLACEMENTS FOR TERMINAL INSTRING AND OUTSTRING MON 503 & 504
125261
125261    DISP 44
125261           DOUBLE                 OSTRADR               % ADDRESS OF STRING DESCRIPTOR IN OUTSTRING
125261    PSID
125261
125261
125261    DISP 46; DOUBLE ISTRADR; PSID                         % STRING ADDRES IN ND-500 INSTRING
125261    DISP 54
125261           DOUBLE                 NBRTADR               % ADDRESS OF NEW BREAK TABLE
125261           DOUBLE                 NECHTADR              % ADDRESS OF NEW ECHO TABLE
125261    PSID
125261    DISP 114;
125261           DOUBLE                 5BRKTAB               % NEW BREAK TABLE BIT 0-31
125261           DOUBLE                 5BRK1
125261           DOUBLE                 5BRK2
125261           DOUBLE                 5BRK3
125261           DOUBLE                 5ECHTAB               % NEW ECHO TABLE BIT 0-31
125261           DOUBLE                 5ECH1
125261           DOUBLE                 5ECH2
125261           DOUBLE                 5ECH3
125261           DOUBLE                 11DMAXBYT             % MAX. NO. OF BYTES FOR MON 511
125261           DOUBLE                 11NOCHRET
125261    PSID
125261
125261    DISP 101; INTEGER NEWPRIO; PSID                       % PARAMETER TO MON 507
125261
125261    %       DISPLACEMENTS FOR MON 505 GET ERROR CODE.
125261
```

```
125261    DISP 152; DOUBLE ERREG; PSID                          % ERROR CODE REGISTER
125261    DISP 100; DOUBLE ERRCODE; PSID                        % RETURNED ERROR CODE IN MON-505
125261          SYMBOL REGBSZ=200                               % SIZE OF ND-500 REGISTER BLOCK
125261                                                          % IN 16 BITS WORDS.
125261
125261    DISP 40
125261          INTEGER 5PPA1,5DPA1,5PPA2,5DPA2,5PPA3,5DPA3,5PPA4,5DPA4
125261          INTEGER 5PPA5,5DPA5,5PPA6,5DPA6,5PPA7,5DPA7,5PPA8,5DPA8
125261    PSID
125261
125261    %       XMSG DISPLACEMENTS IN THE MESSAGE TO THE ND-500
125261
125261    DISP 44; DOUBLE X5BUFADR; PSID
125261    DISP 100
125261          INTEGER AP1, DP1, AP2, DP2, AP3, DP3, AP4, DP4, AP5, DP5, AP6
125261          DOUBLE ADP1=AP1, ADP2=AP2, ADP3=AP3, ADP4=AP4, ADP5=AP5, ADP6=AP6
125261          INTEGER D5TM1=AP4,D5TM2=DP4; DOUBLE D5TIM=D5TM1
125261          INTEGER N5XFUNC=DP1
125261    PSID
125261
125261
125261    DISP 10
125261    INTEGER SWRSTAT
125261    INTEGER SKFLIP
125261    INTEGER PRET1,PRET2,PRET3,PRET4,PRET5,PRET6
125261    INTEGER OLDMICFUNC=PRET1
125261    PSID
125261
125261    DISP 100
125261          DOUBLE RETP1,RETP2,RETP3,RETP4,RETP5,RETP6    % RET.VALUE TO SWAPPER
125261    PSID                                                % AFTER MON. CALLS
125261
125261    DISP 100; DOUBLE SIBNO,SIBSTA; PSID           % SIBAS NUMBER, SIBAS STATUS
125261
125261    %       TRAP STOP REASON:
125261
125261    DISP 12
125261          INTEGER ARRAY         TRAPINFO(40)        % TRAP INFO ARRAY
125261    PSID
125261    DISP 16
125261          INTEGER               TRAPNO              % TRAP NUMBER
125261    PSID
125261
125261
125261    %       STOP REASON CODES:
125261    %
125261          SYMBOL MOCALL=         1                   % MONITOR CALL
125261          SYMBOL TRAPCODE=       2                   % TRAP RECOGNIZED
125261          SYMBOL FATERR=         3                   % ERROR RECOGNIZED IN MICRO-PR
125261
125261
125261    %       SWAPPER FUNCTIONS (SWPFUNC):
125261    %
125261          SYMBOL SWACTIVE=       0                   % SWAPPER ACTIVE
125261          SYMBOL NEWSWAP=        1                   % WANTS TO HANDEL NEW PAGE-FAULT
125261          SYMBOL SWPAGE=         2                   % WANTS TO SWAP IN PAGE
125261          SYMBOL PRSUSPEND=      3                   % SUSPEND PROCESS
125261          SYMBOL ALLOPAGE=       4                   % WANTS TO ALLOCATE NEW PAGE
125261          SYMBOL DATREADY=       5                   % BUFFER READY FROM SWAPPER
125261          SYMBOL CLTSB=          6                   % CLEAR THE TSB OF THE OTHER CPUS
```

```
125261          SYMBOL SWFMAX=CLTSB
125261
125261
125261  %       SUBCOMMANDS TO SWAPPER(SWPSTAT):
125261  %
125261          SYMBOL MSWFIX=          0                       % FIX SEGMENT
125261          SYMBOL MSWUFIX=         1                       % UNFIX SEGMENT
125261          SYMBOL MSWSOVERLAP=     2                       % GIVE INFO ABOUT N500/S III SHARED AREA
125261          SYMBOL MSWMINCR=        3                       % INCREMENT N500 MEMORY
125261          SYMBOL MSWMDECR=        4                       % DECREMENT N500 MEMORY
125261          SYMBOL MSWINIT=         5                       % INITIALIZE SWAPPER
125261          SYMBOL MSWPOUT=         6                       % SWAP OUT ALL PAGES OF PROCESS
125261          SYMBOL MSWSTART=        7                       % START SWAPPER
125261          SYMBOL MSWFORGET=       10                      % FORGET ALL SEGMENTS OF A PROCESS
125261          SYMBOL MSWIPROCESS=     11                      % INIT PROCESS
125261          SYMBOL MSWPFAULT=       12                      % PAGE FAULT
125261          SYMBOL MSWMERR=         13                      % MEMORY ERROR
125261          SYMBOL MSWMCALL=        14
125261          SYMBOL MSWSPRIOR=       15                      % SET PRIORITY OF PROCESS
125261          SYMBOL MSWSGFORGET=     16                      % CLEAR SEGMENT
125261          SYMBOL MSWISEG=         17                      % INIT SEGMENT
125261          SYMBOL MSWRSTART=       20          % RESTART SWAPPER
125261          SYMBOL MSTATISTIC=      21          % SWAPPING STATISTIC
125261          SYMBOL MSWWBACK=        23          % WRITE SEGMENT BACK TO MASS STORAGE
125261          SYMBOL MSWSWAIT=        24          % SET WAIT FOR SWAPPER
125261          SYMBOL MSM510=          27          % MONITOR CALL TO SWAPPER (MON 510)
125261          SYMBOL MSCRS=           30          % CREATE SEGMENT
125261
125261  WICR
125261          SYMBOL                  3WFMPBRK,3RMICV,3RSTAT, 3RWCS,  3WWCS,
125261                                  3SWMESS,3EXAD,   3DEPD,  3RMED,  3WMED,
125261                                  3CACHE, 3RAMD,   3WAMD,  3RESO,  3EXARG,
125261                                  3DEPRG, 3RREG,   3WREG,  3MACLE, 3START,
125261                                  3MONCO, 3TRACO,  3WMONCO,3STAMIC,3PHSREAD,
125261                                  3PHSWRITE,3EXAP, 3DEPP,  3RMEP,  3WMEP,
125261                                  3FQUEUE,3RAMP,3WAMP,  3RLIM,  3PRTRP,
125261                                  3WLIM,  3RPREG,MICFMAX;
125261  WCR;
125261
125261
125261  DISP 25; INTEGER S25; PSID
125261
125261  SYMBOL XMSIZE=120
125261  SYMBOL SWMIN=300,SWMAX=310,FALSE=0,TRUE=1
125261  SYMBOL RIOM=375, CERN=376, N5SWAP=377
125261  SYMBOL 5SEMSIZE=46              % SIZE OF PROCESS DESCRIPTION
125261  SYMBOL TRPINFOSIZE=60           % SIZE OF TRAP INFO IN MESSAGE (BYTES)
125261  SYMBOL 5ETRPINFO=40              % FIRST FREE SPACE IN MESSAGE TO GENERAL DATA
125261  SYMBOL MBUMAX=5MESSIZE-5ETRPINFO  % SIZE OF  FREE SPACE IN MESSAGE TO GENERAL DATA
125261  SYMBOL 5DFSIZE=50               % SIZE OF N500 DATAFIELD TO WRITE ONTO DATA SEG.
125261  SYMBOL 55DFSIZE=5DFSIZE+5DFSIZE
125261
125261  %        CHARACTERISTICS OF MON-60 FUNCTIONS  (BITS IN PARANT)
125261  %        BIT 0-2 ARE NUMBER OF PARAMETERS OF EACH FUNCTION
125261  SYMBOL BM2SEGM=3                % SET IF SG2500 IS THE REENTRAN SEGMENT
125261  SYMBOL RTPLEGAL=4               % COMMAND ALLOWED FROM RT-PROGRAMS
125261  SYMBOL BSPRES=5                 % SYSTEM MUST BE RESERVED FOR SPECIAL USE
125261  SYMBOL COMPROT=6                % COMMAND ONLY ALLOWED
125261                                  % WHEN SYSTEM RESERVED FO SPECIAL USE BY YOU
125261                                  % WHEN USER SYSTEM CALLS
```

```
125261                                  % WHEN AUTO-COMMAND MODE
125261    SYMBOL MULTKVASI=7            % IF BIT MULTKVASI IN PARANT(FUNCTION) THEN
125261                                  % COMMAND LEGAL WHEN NO-PROCESS-TO-COMM.-WITH
125261
125261    SYMBOL COMAUTO=17             % COMMAND IN AUTO-MODE
125261    SYMBOL NSYVARIABLES=70        % NUMBER OF "SYSTEM-DEPENDANT" VARIABLES
125261    SYMBOL BNSYVARIABLES=NSYVARIABLES+NSYVARIABLES
125261    SYMBOL NREGS=65               % NUMBER OF N500 REGISTERS
125261    SYMBOL PRSELSIZE=142          % PROCESS TABLE ELEMENT SIZE
125261    SYMBOL PHSELSIZE=61           % PHYSICAL SEGMENT TABLE ELEMENT SIZE
125261    SYMBOL 5PRSELSIZE=227         % N500 PROCESS TABLE ELEMENT SIZE
125261    SYMBOL 5SSEGSIZE=44            % N500 SEGMENT TABLE SIZE
125261    SYMBOL DSEGSIZE=12000         % SIZE OF DATA SEGMENT
125261    SYMBOL DASEGSTART=110000      % START ADDRESS OF DATA SEGMENT
125261    SYMBOL RSEGSTART=DASEGSTART+DSEGSIZE  % START OF REENTRANT MONITOR SEGMENT
125261    SYMBOL DSSYVARIABLES=RSEGSTART-NSYVARIABLES % START ADDR. OF "SYSTEM-DEPENDANT"
125261                                            % VARIABLES ON DATA SEGMENT.
125261    SYMBOL FT500=DASEGSTART+2     % FLAG-WORD TO N500
125261    SYMBOL FF500=FT500+2          % FLAG-WORD FROM N500
125261    SYMBOL S500DF=FF500+2         % ADDRESS OF DATAFIELD-COPY ON DATA SEGMENT
125261    SYMBOL EXMONC=S500DF+5DFSIZE % ADDRESS OF CODE ON DATA SEGMENT
125261    SYMBOL MCKSIZE=45             % SIZE OF CODE ON DATA SEGMENT
125261    SYMBOL SBUFFER=EXMONC+MCKSIZE
125261    SYMBOL SG1500=30              % ND500 REENTRANT MONITOR SEGMENT 1
125261    SYMBOL SG2500=31              % ND500 REENTRANT MONITOR SEGMENT 2
125261    SYMBOL L12MIN=                500       % FIRST MON CALL THAT REQUIRES SPECIAL
125261                                            % TREATMENT ON LEVEL 12
125261    SYMBOL L12MAX=                514       % LAST
125261    SYMBOL PRSWITCH=              502       % PROCESS SWITCH MONITOR CALL
125261    SYMBOL 5COMPRIOR=             71        % ND-500 COMMUNICATION PRIORITY
125261    SYMBOL 5BRKPRIOR=             55        % BREAK-PRIORITY
125261    SYMBOL HIMSIZE=               15        % SIZE OF HISTOGRAM MESSAGE
125261
125261    % SYMBOLS FOR FAST UDMA INTERFACE
125261
125261    SYMBOL MFANT= 40       % NUMBER OF FIX-ELEMENTS PER PROCESS
125261    SYMBOL MFSIZE=4        % FIX-ELEMENT SIZE
125261    SYMBOL MFNCP= 40       % NUMBER OF DATA CAPABILITIES PER PROCESS
125261
125261    * 5LREG=DASEG
125261    * 5AREG=5LREG
125261    * FPT2E=RSEGS+3               % ENTRY POINT AFTER STANDARD SEGMENT HEADER
125261    * DBASE=FPT2E+2                % ADDR OF BASE FIELD AFTER JPL I *1;5FPT2E
125261    * KEXMC=DBASE+1                % ADDR OF EXECUTE MON CALL ROUTINE THEREAFTER
125261    * SGBRK=DASEG+1
125261
125261
125261
125261    % ************************************************************************
125261
125261    %       N D 5 0 0   D A T A F I E L D   D E F I N I T I O N S
125261
125261    % ************************************************************************
125261
125261    DISP -33
125261    INTEGER TIM1LINK,TIM2LINK
125261    DOUBLE TIMLINK=TIM1LINK       % START OF ND-500 TIME-QUEUE
125261    INTEGER NPPGUWIP              % NUMBER OF PAGES IN PGU/WIP/WIP TABLE
125261    DOUBLE STEXQ                  % START OF THE EXECUTION QUEUE
125261    INTEGER ISTSWPQ
```

```
125261    INTEGER DSTSWPQ
125261    DOUBLE STSWPQ=ISTSWPQ          % START OF THE SWAPPING QUEUE
125261    INTEGER NCPU                   % NUMBER OF CPU'S.  IF DISPLACEMENT IS CHANGED,
125261                                   % ALSO REMEMBER TO CHANGE M5CPU
125261    INTEGER SWOWN                  % CPU DATA FIELD OF THE CPU WHERE SWAPPING IS ALLOWED
125261    INTEGER SYSINITFLG             % INITIALIZIATION FLAGS
125261          SYMBOL BPZEROOK=1        % CONTEXT BLOCKS ALLOCATED
125261          SYMBOL BMDEFOK=2         % MEMORY CONFIGURATION IS OK
125261          SYMBOL BSWLOAD=3         % SWAPPER IS PLACED
125261          SYMBOL BFIRSTACCESS=4    % VERY FIRTST TIME THE MONITOR IS CALLED
125261          SYMBOL BSWFDEF=5         % SWAP-FILE DEFINED
125261          SYMBOL BSUNFIX=6         % UNFIX PAGES AFTER EACH FILE TRANSFER
125261          SYMBOL BSWSTART=7        % SET WHEN THE SWAPPER IS STARTED
125261    INTEGER ZADLINK                % MEMORY MAP ELEMENT ADDR OF N500 PAGE ZERO (IN S-III MEMORY MAP)
125261    INTEGER PHSLINK                % MEMORY MAP LINK FOR SWAPPER:DSEG
125261    INTEGER AN5PAGES(2)            % ACTUAL NUMBER OF PAGES AVAILABLE FOR N500
125261    INTEGER RELBFLG                % SET TO 1 WHILE RELEASING TRANSFER BUFFERS
125261    INTEGER CPUMASK                % IF CPU NO X IS ACTIVE THE
125261                                   % THE CORRESPONDING BIT IS SET
125261    INTEGER NNAMSEG                % SEGMENT NUMBER OF NAME SEGMENT
125261    INTEGER CCPUDF                 % CURRENT CPU DATA FIELD
125261    INTEGER GMAGNO                 % GLOBAL MAGIC NUMBER (CYCLIC PROCESS NR)
125261    INTEGER 5BULINK                % MEMEORY MAP FOR N500 BUFERS
125261    INTEGER 5BUBANK                % MEMORY BANK OF N500 BUFFERS
125261    INTEGER 5BUSTART               % MEMORY ADDR OF N500 BUFFERS
125261          DOUBLE DBUSTART=5BUBANK
125261    INTEGER ADS500S                % ADDRESS OF PROCESS DESCRIPTIONS
125261    INTEGER 5ATM1,5ATM2
125261          DOUBLE 5ATIME=5ATM1      % COPY OF SINTRAN III ATIME
125261    INTEGER SAMPUNITS              % BASIC TIME UNITS BETWEEN EACH SAMPLE
125261    INTEGER PASSIVE                % ALLOWED "PASSIVE" TIME
125261    %       INTEGER                RESLINK,RTRES,BWLINK,TYPRING,ISTATE,MLINK,MFUNC
125261    %       INTEGER POINTER        TRLREG
125261    PAID
125261
125261    %
125261    %       DISPLACEMENTS 0 TO 17 ARE STANDARD MONITOR CALL WORKING FIELD
125261    %
125261
125261    DISP 20
125261    INTEGER 5FUNCTION              % MONITOR CALL FUNCTION CODE
125261    INTEGER C500SEG                % DATA SEGMENT OF CURRENT PROGRAM
125261    INTEGER 5SEMAPHORE             % ADDRESS OF CURRENT PROGRAM'S N-500 SEMAPHORE
125261    INTEGER BUADR                  % MEMORY BANK OF BUFFER
125261    INTEGER 5BUADR                 % PHYSICAL ADDRESS OF BUFFER
125261          DOUBLE DBUADR=BUADR
125261    INTEGER LOGBADR                % LOGICAL ADDRESS OF DEVICE BUFFER
125261    INTEGER 5OLDSEG                % INITIAL SEGMENTS OF CALLING PROGRAM
125261    INTEGER 5RSEGM                 % REENTRAN SEGMENT OF CALLING PROGRAM
125261    INTEGER 5SBITMAP(10)           % BITMAP OF CALLING PROGRAM (BITMAP IN RT.DESCRIPTION)
125261    INTEGER 5D11,5D12              % 2ND. PARAMETER IN MONITOR CALL (32 BITS)
125261    INTEGER 5P1                    % ADDRESS OF 2ND. PARAMETER
125261    INTEGER 5D21,5D22              % 3RD. PARAMETER IN MONITOR CALL (32 BITS)
125261    INTEGER 5P2                    % ADDRESS OF 3RD. PARAMETER
125261    INTEGER 5D31,5D32              % 4TH. PARAMETER IN MONITOR CALL (32 BITS)
125261    INTEGER 5P3                    % ADDRESS OF 4TH. PARAMETER
125261    INTEGER 5D41,5D42              % 5TH. PARAMETER IN MONITOR CALL (32 BITS)
125261    INTEGER 5P4                    % ADDRESS OF 5TH. PARAMETER
125261    INTEGER 5D51,5D52              % 6TH. PARAMETER IN MONITOR CALL (32 BITS)
125261    INTEGER 5P5                    % ADDRESS OF 6TH. PARAMETER
```

```
125261    INTEGER CNTXPAGE              % PAGE NUMBER FOR N500 CONTEXT BLOCK
125261    INTEGER ADRZERO               % N100 PAGE OF N500 ADDRESS ZERO
125261    INTEGER ARRAY AMEMTABLE(20)   % PHYSICAL MEMORY TABLE
125261    INTEGER ARRAY TYPMTAB(10)     % PHYSICAL MEMORY TYPE TABLE
125261    INTEGER NPHSEG                % NUMBER OF PHYSICAL SEGMENTS
125261    INTEGER PGUINTV               % CLEAR PGU INTERVAL IN PAGE FAULTS
125261    INTEGER TOOUTSW               % SET AS OUTSWAP CANDIDATE INTERVAL IN PAGE FAULTS
125261    %INTEGER D1PRIOR               % DEFAULT HIGH PRIORITY IN SINTRAN III WHEN
125261    %                              % WHEN EXECUTING MON CALLS FROM ND-500
125261    %INTEGER D5PRIOR               % DEFAULT PRIORITY IN ND-500
125261    %INTEGER D1CPUPERCENT          % MAX CPU PERCENT ALLOWED ON "HIGH PRIORITY"
125261    %                              % WHEN EXECUTING MON CALL FROM ND-500
125261    INTEGER DCBSZ                 % DISK CACHE BUFFER SIZE
125261    INTEGER DCNBUF                % NUMBER OF DISK CACHE BUFFERS
125261    %INTEGER LPRFAC                % LOW PRIORITY FACTOR
125261    INTEGER MAXFIX                % MAX NUMBER OF PAGES FIXED
125261    %INTEGER NSLPRI                % NUMBER OF SECONDS ON LOW PRIORITY
125261    INTEGER ARRAY XSYSPARS(5)
125261    INTEGER ARRAY SYSPAR=NPHSEGR % SYSTEM PARAMETERS
125261    INTEGER ARRAY FSYVARIABLE(NSYVARIABLES) % "SYSTEM DEPENDANT ADDRESS"
125261    PSID
125261
125261
125261    % ============================================================================
125261    %
125261    %       C P U     S P E C I F I C     D A T A     F I E L D
125261    %
125261    %============================================================================
125261    SYMBOL S5CPUDFSZ=45
125261
125261    DISP -14      % IF THIS IS CHANGED ALSO REMEMBER TO CHANGE THE DEFINITION OF S5CPUDF & E5CPUDF
125261            INTEGER                 CPUNO                 % CPU NUMBER (FIRST NUMBER IS 1)
125261            INTEGER                 WATCHDOG              % ADDRESS OF ND-500 MSG USED BY THE TIMER
125261            INTEGER                 DMLLIM                % LOWER LIMIT FOR INTERFACE
125261            INTEGER                 DMULIM                % UPPER LIMIT FOR INTERFACE
125261            INTEGER                 MIFLAG                % MICROPROGRAM FLAG
125261                  SYMBOL WSMC=0                           % WRITE STRING MONITOR CALLS
125261            INTEGER                 VMICP                 % MICROPROGRAM VERSION
125261    %       INTEGER POINTER         TMSUB
125261    %       INTEGER                 TMR,TTMR
125261    %       INTEGER                 HDEV
125261    %       INTEGER POINTER         STDRIV,DRIVER
125261    %       INTEGER                 RESLINK,RTRES,BWLINK,TYPRING,ISTATE,MLINK,MFUNC
125261    %       INTEGER POINTER         TRLREG
125261    PSID
125261
125261    DISP 7
125261    INTEGER SPREF                 % RT-DESCRIPTION OF PROGRAM RESERVED N500 FOR SPECIAL USE
125261    INTEGER INITFLAG              % =0; N500 NOT INITIALIZED
125261            SYMBOL BCSLOADED=0    % CONTROL STORE IS LOADED
125261            SYMBOL BRESPLACE=10   % RESIDENT-PLACE MODE
125261    INTEGER CMASK                 % BIT NO = CPU NO IS SET
125261    INTEGER TIMSLICE              % TIMESLICE
125261    INTEGER FERROR                % FATAL ERROR IN SYSTEM
125261    INTEGER C5PROC                % ACTIV N500 PROCESS (-1=IDLE)
125261    INTEGER C5STAT                % N-500 STATUS
125261            SYMBOL BHPFAIL=0
125261            SYMBOL BCSLPFAIL=1    % LOADING CS AFTER POWER FAIL
125261    INTEGER  TMRXQ                % ADDRESS OF ERROR-ANSW
125261    INTEGER  MESS1LINK
```

```
125261    INTEGER  MESSLINK                  % TRAVERSE LOC
125261          DOUBLE           MSLNK=MESS1LINK
125261    INTEGER MAIL1LINK
125261    INTEGER MAILINK                    % MESSAGE QUEUE HEAD
125261          DOUBLE            MAILLINK=MAIL1LINK
125261    INTEGER LTTMR                      % EXECUTE TIMEOUT
125261    INTEGER 5PGLINK                    % START OF MEMORY MAP LINK FOR NORD 500 MEMORY
125261    INTEGER 5RESMEMORY                 % MEMORY MAP OF RESIDENT PAGES
125261    INTEGER MPBREAK                    % MICRO-PROGRAM BREAK INFO
125261          SYMBOL MPBENABLE=17          % MICRO-PROGRAM BREAK ENABLED
125261          SYMBOL MPBRWAIT=16           % WAITING FOR MICRO-PROGRAM BREAK
125261          SYMBOL MPBRREACHED=15        % MICRO-PROGRAM BREAK IS REACHED
125261    INTEGER CPUAVAILABLE               % ><0 WHEN ND-500 CPU IS PRESENT
125261    INTEGER C5PWF                      % POWER FAIL FLAG
125261    INTEGER PREVMESS=MPBREAK
125261    TRIPLE 5FRSEGM=5RSEGM
125261    DOUBLE 5DD1=5D11,5DD2=5D21,5DD3=5D31,5DD4=5D41,5DD5=5D51
125261
125261    P5ID
125261    % ***********************************************************************
125261
125261    %     PROCESS DESCRIPTION:
125261
125261    % ***********************************************************************
125261    %
125261    %     THE PROCESS DESCRIPTION KAN BE USED AS A NORMAL DATAFIELD (E.G. LINKED
125261    %     THROUGH THE RESLINK FIELD). ONE IS RESERVED BY THE USER ENTERING THE
125261    %     ND-500-MONITOR.
125261    %     THE PROCESS DESCRIPTIONS ARE KEPT IN A TABLE POINTED TO BY ADS500S IN
125261    %     THE ND-500 DATAFIELD
125261    %
125261    %
125261
125261
125261
125261    DISP 4
125261          INTEGER              PSTAT(3)           % PROCESS STATUS
125261                SYMBOL         5LTSLPRI=10        % TIMESLICED PROCESS ON LOWER TIMESLICE-CLASS PRIORITY
125261                SYMBOL         BSUSPEND=11        % SUSPENDED PROCESS
125261                SYMBOL         BSUSWAIT=12        % SUSPENDED PROCESS IN EX-QUEUE
125261                SYMBOL         SLICE= 013         % TIMESLICED PROCESS
125261                SYMBOL         SOFFLOG=14         % PROCESS LOGGED OFF BY SYSTEM
125261                SYMBOL         5BRK=015           % ESCAPE/BREAK TYPED
125261                SYMBOL         5REWA=016          % WAITS FOR RESTART
125261                SYMBOL         5SVSABORT=17       % PROCESS ABORTED BY SYSTEM
125261          INTEGER              5SEG=PSTAT
125261    %     INTEGER              MLINK              % PART OF STANDARD DATAFIELD
125261    %     INTEGER              MFUNC              % PART OF STANDARD DATAFIELD
125261          INTEGER              MESSBUFF           % PROCESS' BUFFER
125261          INTEGER              T5BUFF             % MESSAGE TO NORD-500 MICRO-PGM
125261          INTEGER              F5BUFF             % MESSAGE FROM NORD-500 MICRO-PGM
125261          INTEGER              H500CPU,L500CPU    % N-500 CPU-TIME USED
125261                DOUBLE         500TU=H500CPU
125261          INTEGER              H100CPU,L100CPU    % N-100 CPU-TIME USED IN PROC.
125261                DOUBLE         N100CPU=H100CPU
125261          INTEGER              5PRIORITY           % PRIORITY OF THE PROCESS
125261          INTEGER              CDTSEGN            % FILE-TRANSFER SEG. NUMBER
125261                                                  % BIT 8-9: RT PROGRAMS ORIGINAL PROTECT RING
125261                                                  % BIT 8-15: FLAGS
125261                SYMBOL         52ESCSET=17        % ESCAPE/BREAK TYPED
```

```
125261                SYMBOL            55REP= 16
125261                SYMBOL            5XBRK=5BRK              % SAVED 5BRK WHILE USING SWAPPER
125261                SYMBOL            OFLDUNLOCK=14           % UNLOCK OFLOCK IF NOT SET
125261                SYMBOL            SPROK=13                % SET PRIORITY MONITOR CALL IS LEGAL
125261          INTEGER                 PLIST(0)                % MONITOR CALL PARAMETER LISTE
125261          INTEGER PMP1,PMP2,PMP3,PMP4,PMP5
125261          INTEGER                 MP1,MP2,MP5,MP3,MP4
125261                DOUBLE            DMP=MP2
125261          INTEGER                 5RUNSTATUS              % IDLE, ACTIVE, INMCALL, USRBRK
125261                SYMBOL 5IDLE=0                             % N500 PROCESS IDLE
125261                SYMBOL ACTIVE=1                          % N500 PROCESS IS ACTIVE
125261                SYMBOL INMCALL=2                         % N500 PROCESS EXECUTES MON CALL
125261                SYMBOL USRBRK=3                          % BREAK; RETURN TO SINTRAN III
125261                SYMBOL 5INCOMM=4                           % IN COMMUNICATION WITH N500
125261                INTEGER           5TRG=PMP1               % T-REG WHEN CALLING FILSYS
125261                INTEGER           5ARG=PMP2               % A-REG WHEN CALLING FILSYS
125261                INTEGER           5DRG=PMP3               % D-REG WHEN CALLING FILSYS
125261                INTEGER           5XRG=PMP4               % X-REG WHEN CALLING FILSYS
125261                INTEGER           5ERRFLAG=PMP5           % ERROR INDICATOR WHEN CALLING FILSYS
125261                INTEGER           5BRG=5TRG
125261                INTEGER POINTER   FSYSENTRY=MP1           % ADDRESS OF FILSYS ENTRY POINT
125261                INTEGER POINTER   LRET=FSYSENTRY          % RETURN ADDRESS IN GDSEG/GNSEG
125261                TRIPLE            5TADRG=5TRG
125261                INTEGER           5CURSEGS=MP1            % OLD SEGMENTS IN RSGMONC
125261          INTEGER                 HTSLLOWPRI              % HIGEST "LOW-TIMESLICE-CLASS" PRIORITY
125261          INTEGER                 HOLDCPU,LOLDCPU
125261                DOUBLE            OLDCPU=HOLDCPU          % CPU-TIME USED AT LAST SAMPLE POINT
125261          INTEGER                 CRSEGM                  % CURRENT REENTRANT SEGMENT
125261          INTEGER                 PDCFLG                  % USED FOR FLAGGING ABORTING BY SYSTEM ON ERROR CONDITION
125261          INTEGER                 SV5FUNC                 % USED TO SAVE CURRENT FUNCTION IN MON 60
125261          INTEGER                 CPUPRCNT                % N500 USE OF THE N100 CPU
125261                                                          % MUST NOT EXCEED THS PERCENT
125261                INTEGER SUSPMESSTAT=CPUPRCNT              % ORIGINAL STATUS WHEN SUSPENDED
125261          INTEGER                 LPRCOUNT                % FOR SPECIAL TIME COUNTING
125261          INTEGER                 MAGNO                   % "MAGIC" PART OF THE PROCESS NO
125261          INTEGER                 OUTDF                   % ADDRESS OF TERMINAL OUTPUT DATA FIELD
125261                INTEGER            5DUNIT= PMP1
125261                INTEGER            5LUNIT= PMP2
125261                INTEGER            5NFLAG= PMP3
125261                INTEGER            1PAVA=  PMP4
125261                INTEGER            2PAVA=  PMP5
125261                INTEGER            5SECTO= MP1
125261          INTEGER                 CPUDF                   % CPU DATA FIELD OF THE CPU
125261                                                          % THAT THIS PROCESS IS RUNNING ON
125261    PSID
125261
125261    % ************************************************************************
125261
125261    %       IOX DEFINITIONS
125261
125261    % ************************************************************************
125261
125261
125261          SYMBOL                  RMAR= 000               % READ MEMORY ADDRESS REG.
125261          SYMBOL                  LMAR= 001               % LOAD MEMORY ADDRESS REG.
125261          SYMBOL                  RSTA= 002               % READ STATUS REGISTER
125261          SYMBOL                  LSTA= 003               % LOAD STATUS REGISTER
125261          SYMBOL                  RCON= 004               % READ CONTROL REGISTER
125261          SYMBOL                  LCON= 005               % LOAD CONTROL REGISTER
125261          SYMBOL                  5MCLR= 006               % NORD-500 MASTER CLEAR
```

```
125261          SYMBOL                  TERMI= 007              % NORD-500 TERMINATE
125261          SYMBOL                  RTAG= 010               % READ TAG
125261          SYMBOL                  RUPP= RTAG              % READ UPPER LIMIT REGISTER
125261          SYMBOL                  LTAG= 011               % LOAD TAG
125261          SYMBOL                  LUPP= LTAG              % LOAD UPPER LIMIT REGISTER
125261          SYMBOL                  RLOW= 012               % READ LOW LIMIT REGISTER
125261          SYMBOL                  LDAT= 013               % LOAD DATAX
125261          SYMBOL                  LLOW= LDAT              % LOAD LOW LIMIT REGISTER
125261          SYMBOL                  SLOC= 014               % LOCK NORD-500 COMMUNICATION
125261          SYMBOL                  CLKD= 015               % CLOCK DATA
125261          SYMBOL                  UNLC= 016               % UNLCK NORD-500 COMMUNICATION
125261          SYMBOL                  RETG= 017               % RETURN TAG
125261
125261  %       NORD-500 INTERFACE STATUS REGISTER BITS:
125261
125261          SYMBOL                  5ACTIV=2                % N-500 ACTIVATE
125261          SYMBOL                  5READY=3                % N-500 READY FOR TRANSFER
125261          SYMBOL                  5PAGF=4                 % ERROR ON THE INTERFACE
125261          SYMBOL                  5ILOCK=5                % INTERFACE LOCKED
125261          SYMBOL                  5DMAER=6                % DMA-ERROR
125261          SYMBOL                  5PFAIL=7                % POWER FAIL EXECUTED BY MIC-P.
125261          SYMBOL                  5POWOF=10               % POWER OFF
125261          SYMBOL                  5CLOST=11               % NORD-500 CLOCK STOPPED
125261
125261
125261
125261  %*******************************************************************************
125261  %
125261  %               F U N C T I O N S  I N  M O N I T O R  C A L L  6 0
125261  %
125261  %*******************************************************************************
125261  SYMBOL RREG=                    0               % READ REGISTER
125261  SYMBOL WREG=                    1               % WRITE REGISTER
125261  SYMBOL PMREAD=                  2               % READ PROGRAM MEMORY
125261  SYMBOL DMREAD=                  3               % READ DATA MEMORY
125261  SYMBOL PMWRITE=                 4               % WRITE PROGRAM MEMORY
125261  SYMBOL DMWRITE=                 5               % WRITE DATA MEMORY
125261  SYMBOL SEGLOAD=                 6               % LOAD SEGMENT
125261  SYMBOL PLSWAPPER=               7               % PLACE SWAPPER
125261  SYMBOL RREGS=                   10              % READ REGISTERS
125261  SYMBOL WREGS=                   11              % WRITE REGISTERS
125261  SYMBOL PRSTART=                 12              % START PROGRAM
125261  SYMBOL FILCON=                  13              % CONNECT FILE
125261  SYMBOL FILCLO=                  14              % CLOSE FILE
125261  SYMBOL N5RES=                   15              % ALLOCATE NORD 500 PROCESS
125261  SYMBOL N5REL=                   16              % RELEASE NORD 500 PROCESS
125261  SYMBOL FLIOP=                   17              % LIST OPEN FILES
125261  SYMBOL TIMEUS=                  20              % TIME USED
125261  SYMBOL WISON=                   21              % WHO IS ON
125261  SYMBOL ERRFSET=                 22              % SET ERRFLAG
125261  SYMBOL RCNTS=                   23              % READ CONTROL STORE
125261  SYMBOL WCNTS=                   24              % WRITE CONTROL STORE
125261  SYMBOL MICPSTART=               25              % MICRO PROGRAM START
125261  SYMBOL DMEXA=                   26              % DATA MEMORY EXAMINE
125261  SYMBOL DMDEP=                   27              % DATA MEMORY DEPOSIT
125261  SYMBOL PMEXA=                   30              % PROGRAM MEMORY EXAMINE
125261  SYMBOL PMDEP=                   31              % PROGRAM MEMORY DEPOSIT
125261  SYMBOL DAMR=                    32              % ABSOLUTE DATA MEMORY READ
125261  SYMBOL DAMW=                    33              % ABSOLUTE DATA MEMORY WRITE
125261  SYMBOL STOPMIC=                 34              % STOP MICRO PROGRAM
```

```
125261    SYMBOL FMCLEAR=          35      % MASTER CLEAR
125261    SYMBOL ALLPSEG=          36      % ALLOCATE PROGRAM SEGMENT
125261    SYMBOL CSLOAD=           37      % LOAD CONTROL STORE
125261    SYMBOL MEMDEF=           40      % DEFINE MEMORY CONFIGURATION
125261    SYMBOL RSTATU=           41      % READ N500 AND N100 COMMUNICATION STATUS
125261    SYMBOL ABREL=            42      % ABORT CURRENT PROCESS AND ND-100 PROGR.
125261    SYMBOL N5SRS=            43      % RESERVE N500 FOR SPECIAL USE
125261    SYMBOL N5SRL=            44      % RELEASE N500 FROM SPECIAL USE
125261    SYMBOL SCPLOOP=          45      % SCOPE-LOOP
125261    SYMBOL DEFSWAP=          46      % DEFINE-SWAP-FILE
125261    SYMBOL DELSWAP=          47      % DELETE-SWAP-FILE
125261    SYMBOL TSTFUNC=          50      % TEST-FUNCTION
125261    SYMBOL RIREG=            51      % READ INTERFACE (COMMUNICATION) IODATUT REG.
125261    SYMBOL GIVPAGES=         52      % GIVE N500 SWAPPING PAGES
125261    SYMBOL TAKPAGES=         53      % TAKE N500 SWAPPING PAGES
125261    SYMBOL STSWAPPER=        54      % START SWAPPER PROCESS
125261    SYMBOL SPLACE=           55      % START PLACE
125261    SYMBOL EPLACE=           56      % END PLACE
125261    SYMBOL RMVERS=           57      % READ MICROPROGRAM VERSION
125261    SYMBOL LIMEM=            60      % LIST MEMORY CONFIGURATION
125261    SYMBOL MRESSPES=         61      % RESERVE N500 MEMORY FOR SPECIAL USE
125261    SYMBOL 5DEFHIST=         62      % DEFINE HISTOGRAM
125261    SYMBOL 5STAHIST=         63      % START HISTOGRAM
125261    SYMBOL 5STOHIST=         64      % STOP HISTOGRAM
125261    SYMBOL REAHIST=          65      % READ HISTOGRAM
125261    SYMBOL RELHIST=          66      % RELEASE HISTOGRAM
125261    SYMBOL SPRTE=            67      % SEARCH FOR PROCESS ENTRY ON NAME SEG
125261    SYMBOL GPRTE=            70      % READ PROCESS ENTRY FROM NAME SEG
125261    SYMBOL SSGTE=            71      % SEARCH FOR PH. SEG. ENTRY ON NAME SEG
125261    SYMBOL GSGTE=            72      % READ PH. SEG. ENTRY FROM NAME SEG
125261    SYMBOL RPHSG=            73      % READ PHYSICAL SEGMENT
125261    SYMBOL SPRNM=            74      % SET PROCESS NAME OF CURRENT PROCESS
125261    SYMBOL TSTUSER=          75      % TEST FOR USER SYSTEM
125261    SYMBOL TOSWP=            76      % SEND MESSAGE TO SWAPPER
125261    SYMBOL RMESSAGE=         77      % READ LAST MESSAGE
125261    SYMBOL 5RFLAG=           100     % READ FLAG-INFO FROM N500
125261    SYMBOL 5WFLAG=           101     % WRITE FLAG-INFO TO N500
125261    SYMBOL FORGET=           102     % RELEASE N500 SYSTEM FROM SINTRAN III
125261    SYMBOL RSYSP=            103     % READ SYSTEM PARAMETERS
125261    SYMBOL WSYSP=            104     % WRITE SYSTEM PARAMETERS
125261    SYMBOL S1PRIOR=          105     % SET PRIORITY OF BACKGROUND PROGRAM
125261    SYMBOL LINKTO=           106     % LINK TO PROCESS
125261    SYMBOL MICBRK=           107     % MICRO PROGRAM BREAK
125261    SYMBOL WPHSG=            110     % WRITE INTO PHYSICAL SEGMENT
125261    SYMBOL 5STAPRLOG=        111     % START PROCESS LOG
125261    SYMBOL 5STOLOG=          112     % STOP LOGGING
125261    SYMBOL PRILOG=           113     % PRINT LOGG-INFO
125261    SYMBOL RELLOG=           114     % RELEASE LOGGING
125261    SYMBOL STLAPROC=         115     % START LOGG ALL ACTIVE PROCESSES
125261    SYMBOL XN5REL=           116     % LOGOUT OWN PROCESS
125261    SYMBOL PRSTOP=           117     % ABORT PROCESS
125261    SYMBOL SOUTFIL=          120     % SET PRINT DEVICE
125261    SYMBOL RSWPDATA=         121     % READ FROM SWAPPER'S DATA MEMORY
125261    SYMBOL LOGOFF=           122     % LOGG OFF PROCESS
125261    SYMBOL MRELSPES=         123     % RELEASE ND-500 MEMORY
125261    SYMBOL STAMLOG=          124     % RESERVE AND START MONCALL LOGG
125261    SYMBOL PRIMLOG=          125     % PRINT MONCALL LOGG
125261    SYMBOL STOMLOG=          126     % STOP AND RELEASE MONCALL LOGG
125261    SYMBOL DFSYDOM=          127     % DEFINE SYSTEM-DOMAIN
125261    SYMBOL SFSYDOM=          130     % SEARCH FOR SYSTEM-DOMAIN
```

```
125261    SYMBOL DLSYDOM=                131        % DELETE SYSTEM-DOMAIN
125261    SYMBOL LISYDOM=                132        % LIST "SYSTEM-DOMAINS"
125261    SYMBOL LISEXQU=                133        % LIST ND-500 EXECUTION QUEUE
125261    SYMBOL PLDEB=                  134        % PLACE DEBUGGER
125261    SYMBOL ABLOG=                  135        % LOGG OFF PROCESS AND ABORT PROGRAM
125261    SYMBOL PRACTIVATE=             136        % ACTIVATE STOPPED PROCESS
125261    SYMBOL STENPLACE=              140        % START RESIDENT-PLACE
125261    SYMBOL S5BLSIZE=               141        % SET BLOCK SIZE OF A FILE
125261    SYMBOL S3CPNT=                 142        % VALUE OF BYTE POINTER IN COMMAND BUFFER
125261    SYMBOL MO5RT=                  143        % ACTIVATE ND-500 PROC. OR ND-100 PROG.
125261    SYMBOL CHACPU=                 144        % CHANGE CPU
125261    SYMBOL SSTDOM=                 145        % START STANDARD DOMAIN FROM COMMAND SEG.
125261    % FUNCTION CODE 146 IS USED IN DRIVER/SYSTEM MONITOR
125261    SYMBOL N5S3REL=                147        % ESCAPE TYPED WHEN MONITOR STARTED FROM S3.OPCOM.
125261    SYMBOL LI5TMQ=                 150        % LIST ND-500 TIME-QUEUE
125261    SYMBOL FUNCMAX=LI5TMQ
125261
125261    @CR
125261            INTEGER ARRAY PARANT:=(
125261                  22\22,24\24,23\23,35\151,21\21,23\25,21\32,20\1,
125271                  11\212,2\153,153\151,22\22,22\22,124\123,170\170,13\153,
125301                  153\33,0\130,130\152,111\111,214\331,131\131,150\30,231\11,
125311                  231\132,23\20,20\21,20\112,13\113,13\115,31\10,31\22,
125321                  22\22,130\331,331\34,111\112,134\101,100\102,100\101,0\21,
125331                  11\124,121\130,21\21,21\111,32\111,210\201,32\331,331\0,
125341                  171\2,11\321,21\11,0\0,201\0);
125346    @CR;
125346    *)KILL VENDC; VENDC=*
125346    *)ENDC/
066700
066700
```

```
066700
066700  % ******************************************************************************
066700  %       N 5 0 0 M
066700  % ******************************************************************************
066700
066700
066700  % ENTRY POINT OF MONITOR CALL 60
066700
066700  N500M: CALL GET1; *POF
066702         GO FAR PFN5M; *)FILL
066705  *)KILL 7ENDC; 7ENDC=*
066705  *)ENDC/
125346  PFN5M: IF 5FUNCTION BZERO COMAUTO>>FUNCMAX OR FUNCS(A)=0 GO FAR ILLFUNC
125357         RTREF=:D; CALL FSEMA; GO N5PDRES; GO FAR 5PDRES      % FIND PROCESS DESC IF RESERVED
125364  N5PDRES:                                                    % NO PROCESS DESCRIPTION RESERVED
125364        IF X:=5FUNCTION BZERO COMAUTO=N5REL GO FAR OKRET
125371        IF X=N5RES OR X=MRESSPES OR X=SSTDOM THEN             % TRY TO RESERVE A FREE PROC.DESC
125402           CALL CHIBATCH; GO INPFN5                           % CHECK IF BATCH
125404           FOR X:="S500S+5SEMSIZE" STEP 5SEMSIZE TO "S500E-5SEMSIZE-5SEMSIZE" DO
125410               IF X.RTRES=0 THEN                              % FREE PROC.DESC
125412  INPFN5:          X=:B:=RTREF; CALL BRESERVE                 % RESERVE THE PROCESS DESC.
125415  PRFOUND:         A:=B-"S500S"=:D:=0; T:=5SEMSIZE; *RDIV ST  % COMPUTE PROCESS NUMBER
125423                   A=:"N500DF".5D32; 0=:PDCFLG
125426                   IF X:=X.SWOWN><0 AND X.SPREF=0 AND X.CPUAVAILABLE><0 THEN
125434                      GO CPUOK
125435                   ELSE
125436                      FOR X:="S5CPUDF" STEP 5CPUDFSZ TO "E5CPUDF" DO
125442                          IF X.SPREF=0 AND X.CPUAVAILABLE><0 GO CPUOK
125446                      OD; X:=RTREF; CALL BRELEASE; "N500DF"=:B; GO FAR EESPRES
125455                   FI
125455  CPUOK:           X=:CPUDF; "N500DF"; A:=:B; X=:CCPUDF; A=:5SEMAPHORE
125462                   IF A.5SEG/\377=0 THEN CALL ERRFATAL FI
125467                   A=:C500SEG BONE SLICE=:X.5SEG              % ALL ND-500 PROCS ARE DEFAULT TIMESLICED
125472                   5IDLE=:X.5RUNSTATUS
125474                   5BRKPROR=:X.5PRIORITY; 0=:X.LPRCOUNT
125477                   X.CDTSEGN/\377=:X.CDTSEGN
125502                   0=:X.H500CPU=:X.L500CPU
125504                   5TSLSTATUS(5D32) BONE 5BRKF=:5TSLSTATUS(X)
125510                   GO FAR N5001; *)FILL
125533                FI
125533           OD
125535        FI; X:=5FUNCTION BZERO COMAUTO; T:="PARANT"; *LBYT
125541        IF A NBIT MULTKVASI THEN ENOPCOM; GO FAR ERET FI
125545  XWISON: "S600E"=:B; X:=RTREF; CALL BRESERVE                 % RESERVE THE VERY LAST PROC.DESC.
125551        IF A<0 THEN
125552           CALL FREXQU; CALL TOWQU; "N500DF"=:B; CALL BRELEASE
125557           ZPREG-1=:ZPREG; GO RETSTUPR                        % WAIT UNTIL THE VERY LAST PROC.DESC IS FREE
125563        FI; GO PRFOUND
125564
125564  5PDRES:                                                     % PROC.DESC IS RESERVED
125564        X=:5SEMAPHORE
125565        IF 5FUNCTION BZERO COMAUTO=N5RES THEN X.5SEG BONE SLICE=:X.5SEG ELSE X.5SEG FI
125577        A/\377=:C500SEG; 5SEMAPHORE.CPUDF.SPREF=:D; X=:CCPUDF
125606        IF D><0 AND D><RTREF AND 5FUNCTION><N5REL AND A><ABLOG AND A><M05RT THEN
125625           CALL FSEMA; GO 5PDR1
125627           GO FAR EESPRES                    % ND-500 CPU IS RESERVED FOR SPECIAL USE BY A STILL ACTIVE P
125630  5PDR1:   0=:CCPUDF.SPREF                  % THE PROC. RESERVED THE ND-500 CPU IS NO LONGER ACTIVE
125632           X.INITFLAG BZERO BRESPLACE=:X.INITFLAG
125635        FI; GO FAR N5001
125636  *)FILL
```

```
125655    *)KILL YENDC; YENDC=*
125655    *)ENDC/
066705
066705    ESBUF:    EN5BUFFR; GO ERET
066707    EESPRES:  ESPRES; GO ERET
066711    BUFTBIG:  *BSET ZRO
066712              EBIGBUF; GO ERET
066714    EILPAR:   EILPAR; GO ERET
066716    ILLFUNC:  EILFUNC
066717    ERET:     A=:"N500DF",ZAREG; X=:B; MLEV; *PON; MST PIE
066725              GO FAR RELBUF
066726    *)FILL
066734
066734    INTEGER CFP,CLP,CNP,MXCLP
066740    INTEGER POINTER 5GBLR
066741    INTEGER ARRAY PM61:=("1",CNP,CFP,CLP,"0","0"),PPRT:=(CPRTP)
066750    INTEGER CPRTP=?
066750
066750    % SUBROUTINE TO CHECK THAT THE VARIOUS BUFFERS USED BY THE ND-500 SYSTEM
066750    % NOT WILL BE PLACED IN THE USER-RESERVED MEMORY AREA
066750    CSUBR:  X:=0
066751            DO WHILE X<<"CUMSIZE*2"
066754               X=:CPRTP
066755               A:="CUMTABLE"; X+A; T:=0; *LDDTX
066761               IF A<<=CFP AND D+1>>T THEN T:=D=:CFP FI
066771               IF X:=CFP>>=CLP THEN EXIT FI
066776               IF A<<=CLP AND A>>=CFP THEN A-1=:CLP FI
067006               X:=CPRTP+2
067010            OD; EXIT
067012
067012
067012    % SUBROUTINE TO RESERVE MEMORY IN LOWER 4 MEMORY BANKS
067012    %
067012    % ENTRY:       A=NUMBER OF PAGES TO RESERVE
067012    % EXIT:        NO MEMORY AVAILABLE
067012    % EXIT AD1:    A=FIRST PHYSICAL PAGE RESERVED
067012    %
067012    5GBUFF: 0=:CFP; A=:CNP; T=:MXCLP:=L=:"5GBLR"
067017            DO
067017               CFP/\177700+77=:CLP; CALL CSUBR
067024            WHILE CFP<<MXCLP
067030               "PM61"; *MON 61
067032               GO 5GB2; MIN "5GBLR"; GO 5GBLR
067035    5GB2:      CLP+1=:CFP
067040            OD; GO 5GBLR
067042    *)FILL
067047
067047    % CHECK IF CURRENT PROGRAM IS A BATCH PROCESS
067047    % IF IT IS A BATCH PROCESS AND THE LAST PROCESS DESCRIPTION IS FREE
067047    % THEN THIS SUBROUTINE WILL GIVE SKIP-RETURN ELSE IT WILL GIVE RETURN
067047
067047    CHIBATCH: MLEV; *PON; MST PIE
067052            IF BACKGROUND><0 AND 5BCHFLAG=1 THEN
067060               X:="S500E-5SEMSIZE"
067061               IF X.RTRES><0 THEN L+1 FI
067064            ELSE
067065               L+1
067066            FI; MLEV; *MCL PIE; POF; EXIT
067072    *)FILL
067075
```

```
067075    INTEGER CPRTP
067076
067076    N5001: IF RELBFLG = 1 GO FAR EESPRES
067102           CPUMASK \/ CCPUDF.CMASK=:CPUMASK
067106    INAGN:
067106           IF 2>5MSINIT THEN                                  % ND-500 NOT INITIALIZED?
067112              MLEV; *PON; MST PIE
067115              IF T=0 THEN                                     % T= OLD 5MSINIT
067117                 A:="MSGMX"; D:=0; AD SHZ -12; IF D><0 THEN A+1 FI
067125                 T:=400; CALL 5GBUFF; GO 0E5BUFFR             % RESERVE MEMORY FOR ND-500 MESSAGES
067130                 A=:D
067131                 MLEV; *MCL PIE; POF
067134                 CALL MSINIT                                  % INITIALIZE THE MESSAGE BUFFERS
067135                 MLEV; *PON; MST PIE
067140    * 8N500 8F5UD
067140                  CMXACTPROC-1*"MFANT*MFSIZE+MFNCP"                % ALLOCATE BUFFER FOR FAST MON UDMA INTERFACE
067143                 D:=0; AD SHZ -12; IF D><0 THEN A+1 FI
067150                 T:=ENDPAGE; CALL 5GBUFF; GO 0E5BUFFR
067153                 A=:D SHZ -6=:5FXBNK:=D SH 12=:5DSPS
067161                 CMXACTPROC-1*MFNCP+5DSPS=:5FXTBL
067166                 FOR X:="9FPUD" STEP 5RTSIZE TO "9LPUD-5RTSIZE" DO
067172                    X=:CPRTP; "PPRT"; *MON 100
067175                 OD
067177    * 8N500
067177                  FI; "5SWAP"=:CPRTP; "PPRT"; *MON 100          % START 5SWAP
067203              CMXACTPROC; T:=400; CALL FAR 5GBUFF; GO 1E5BUFFR  % RESERVE MEMORY FOR "MON 60" COMMUNICATION
067207              A=:D;=0; AD SH 12=:DBUSTART
067213              T=:5BULINK; 2=:5MSINIT
067216              A:=-1=:D; AD=:TIMLINK=:STSWPQ                  % CLEAR ND-500 TIME QUEUE AND SWAP QUEUE
067222           FI
067222           GO N5003; *)FILL
067243
067243    0E5BUFFR: 0=:5MSINIT; GO XE5BUFFR
067245    1E5BUFFR: 1=:5MSINIT
067247    XE5BUFFR: GO FAR E5BUFFR
067250
067250
067250    INTEGER XPARANT
067251
067251    N5003: MLEV; *PON; MST PIE
067254           CALL XGETBUFF
067255           X:=5FUNCTION BZERO COMAUTO
067257           T:="PARANT"; *PIOF; LBYT; PION
067263           A=:D/\7=:XPARANT
067266           IF D BIT BM2SEGM THEN SG2500 ELSE SG1500 FI
067273           A=:5SEMAPHORE.CRSEGM=:C5REENT
067276           IF BACKGROUND=0 AND D NBIT RTPLEGAL THEN EFUNRTP; GO FAR ERET FI
067304           IF D BIT COMPROT THEN
067306              IF BACKGROUND><0 AND 5PASSTYPE><2 AND 5FUNCTION NBIT COMAUTO THEN
067317                 ENAUTHORIZED; GO FAR ERET; *)FILL
067333              FI
067333              IF D BIT BSPRES THEN
067335                 IF CCPUDF.SPREF><0 AND A><RTREF THEN ENSPREF; GO FAR ERET FI
067345                 MLEV; *MCL PIE; POF
067350                 CALL TUSON; GO FAR ERET
067352                 MLEV; *PON; MST PIE
067355              FI
067355           FI; 5FUNCTION BZERO COMAUTO=:5FUNCTION
067360           RTREF.ACTPRI/\163777=:X.ACTPRI
067364           IF BACKGROUND><0 THEN
```

```
067366              CALL XOFFESC; X.STATUS BONE 11 BZERO 5REP=:X.STATUS
067373              5BCOM=:5TTIFIELD.BSTATE
067376           ELSE
067377              X.STATUS/\1400=:T; X.STATUS/\176377\/1000=:X.STATUS
067406              5SEMAPHORE.CDTSEGN\/T=:X.CDTSEGN
067412           FI
067412           I:=XPARANT; "5D11"+B=:X; ZAREG+1=:B; CALL 5ALTON
067422           DO WHILE T><0                                        % MOVE PARAMETERS TO THE ND-500 DATAFIELD
067424              *BSET ONE; LDD I,B; BSET ZRO; STD ,X
067430              *BSET ONE; LDA ,B; BSET ZRO; STA 2,X
067434              X+3; B+1; T-1
067437           OD; *BSET ZRO
067441           "N500DF"=:B
067443           MLEV; *MCL PIE; POF
067446           5IFUNC(5FUNCTION)=:P                                 % GO TO ENTRY POINT ACCORDING TO FUNCTION CODE
067451     *)FILL
067470     *)KILL 7ENDC; 7ENDC=*
067470     *ENDC/
125655
125655     % TABLE OF ENTRY POINTS TO GOTO BEFORE CALLING THE SYSTEM MONITOR
125655     % THE FUNCTION CODE IS INDEX IN THE TABLE
125655     $LCR
125655     INTEGER ARRAY 5IFUNC:=(
125655              NOPAR,NOPAR,NOPAR,NOPAR,IPMWRITE,IDMWRITE,ISEGLOAD,IPLSWAPPER,
125665              NOPAR,IWRGS,NOPAR,ICONNFI,NOPAR,NOPAR,NOPAR,NOPAR,
125675              NOPAR,NOPAR,NOPAR,NOPAR,IWCNT,NOPAR,NOPAR,NOPAR,
125705              NOPAR,NOPAR,NOPAR,IDAMW,NOPAR,NOPAR,NOPAR,ICSLOAD,
125715              IMEMDEF,NOPAR,NOPAR,ISRES,ISREL,NOPAR,IDEFSWAP,IDELSWAP,
125725              NOPAR,NOPAR,NOPAR,NOPAR,NOPAR,ISPLACE,IEPLACE,NOPAR,
125735              NOPAR,IMRESSPES,IDEFHIST,ISTAHIST,ISTOHIAT,IREAHIST,IRELHIST,ISPRTE,
125745              NOPAR,ISSGTE,NOPAR,NOPAR,ISPRNM,ITSTUSER,ITOSWP,IRMESS,
125755              RWFLAG,RWFLAG,IFORGET,IRSYSP,IWSYSP,NOPAR,NOPAR,NOPAR,
125765              IWPHSG,ISTAPRLOG,ISTOLOG,IPRILOG,IRELLOG,ISTLAPR,NOPAR,IPRABORT,
125775              NOPAR,NOPAR,ILOGOFF,IMRELSPES,ISTAMLOG,IPRIMLOG,ISTOMLOG,IDFSYDOM,
126005              ISFSYDOM,IDLSYDOM,NOPAR,ILI5EXQ,IPLDEB,IABLOG,IPRACTIVE,0,
126015              NOPAR,NOPAR,NOPAR,IMO5RT,ICHACPU,ISSTDOM,ILLFUNC,ILLFUNC,
126025              ILI5TQU);
126026     $UCR;
126026
126026     % SUBROUTINE TO CHECK IF OTHER USERS ARE LOGGED ON THE ACTUAL ND-500
126026     % IF NONE ARE LOGGED ON THEN RESERVE THE ACTUAL ND-500 FOR SPECIAL USE
126026     % FOR THE CALLING PROGRAM
126026     %
126026     % ENTRY:      NONE
126026     % EXIT:       OTHER USERS ARE LOGGED ON
126026     % EXIT AD1:   THE ACTUAL ND-500 IS RESERVED FOR SPECIAL USE
126026     TUSON: FOR X:="S500S+5SEMSIZE" STEP 5SEMSIZE TO "S500E" DO
126032               IF X.RTRES><0 AND A><RTREF AND X.CPUDF = CCPUDF THEN EUSRON; EXIT FI
126045            OD; RTREF=:CCPUDF.SPREF; EXITA
126053
126053     ISSTDOM: D=:5P1; GO FAR NOPAR
126055     ISRES: IF CCPUDF.SPREF=RTREF GO FAR OKRET
126062            CALL TUSON; GO ISRE1; GO FAR OKRET
126065     ISRE1: IF CCPUDF.SPREF=0 THEN RTREF=:X.SPREF FI
126072            EUSRON; GO FAR ERET
126074     IMRESSPES: IF 5D11 = -1 AND 5D12 <<= NCPU THEN
126104                  IF A><0 THEN A-1 FI
126106                  A*5CPUDFSZ+"S5CPUDF"=:X=:CCPUDF
126112                  IF X.SPREF >< 0 AND A >< RTREF GO FAR EESPRES
126117                  CCPUDF=:5SEMAPHORE.CPUDF
```

```
126122          FI
126122          CALL TUSON; GO FAR ERET; GO FAR NOPAR
126125  IMRELSPES: IF CCPUDF.SPREF><RTREF GO FAR OKRET
126132          GO FAR NOPAR
126133  IGRET:  IF CCPUDF.SPREF><RTREF GO FAR OKRET
126140          0=:X.SPREF; X.INITFLAG BZERO BRESPLACE=:X.INITFLAG; GO FAR OKRET
126145  IWPHSG: AD:=5DD3; IF A><0 OR D>>4000 GO FAR BUFTBIG
126152          T:=5D32; 5P4; CALL CMOVB
126155          GO FAR NOPAR
126156
126156  ICHACPU:                        % CHANGE CPU
126156          IF 5D12 =0 OR A >> NCPU GO FAR EEILPAR    % CHECK THAT CPU EXISTS
126163          A=1*5CPUDFSZ+"S5CPUDF"=:X
126167          IF X.CPUAVAILABLE=0 GO FAR EESPRES
126172          IF X.SPREF >< 0 AND A >< RTREF GO FAR EESPRES
126177          IF A:=X >< 5SEMAPHORE.CPUDF THEN 0=:T.SPREF FI
126206          A=:5SEMAPHORE.CPUDF; GO FAR OKRET; *)FILL
126230
126230  IDMWRITE:
126230  IDMWRITE:
126230  IDAMW:  AD:=5DD1; IF A><0 OR D>>4000 GO FAR BUFTBIG
126235          T:=5D12; 5P3; CALL CMOVB
126240          GO FAR NOPAR
126241  ISEGLOAD:
126241          IF 5D51><0 THEN
126243             LOGBADR+100=:LOGBADR
126246             T:=40; 5P5; CALL CMOVB; *BSET ZRO
126252             LOGBADR-100=:LOGBADR
126255          FI; GO 5COPYFIL
126256  ISPLACE:
126256          5SEMAPHORE.CDTSEGN BZERO 55REP=:X.CDTSEGN; GO FAR NOPAR
126263  IEPLACE:
126263  IWRGS:
126263  IWRG1:  T:=NREGS SH 2; 5P1; CALL CMOVB
126267          GO FAR NOPAR
126270  IWCNTS: IF 5D22>>2000 GO FAR ERET
126274          T:=5D22 SH 1; 5P3; CALL CMOVB
126300          GO FAR NOPAR
126301  IMEMDEF: T:=5D22 SH 2; 5P3; CALL CMOVB
126305          GO FAR NOPAR
126306  *)FILL
126314
126314  IIOSWP: 5P1+7; T:=5MESSIZE-7 SH 1; GO ICOPF
126322  IDFSYDOM: 5P1; T:=4000; GO ICOPF
126325  IPRACTIVE:
126325  ISPRNM:
126325  IGPRNUMBER: 5P1; T:=50; GO ICOPF
126330  IPLDEB:   5P2; GO ICOPYF
126332  IDFSYDOM:
126332  IDLSYDOM:
126332  ISPRTE:
126332  ISSGTE:
126332  IDELSWAP:
126332  IDEFSWAP:
126332  IPLSWAPPER:
126332  5COPYFIL: 5P1
126333  ICOPYF:   T:=200
126334  ICOPF:    CALL CMOVB; GO FAR NOPAR
126336
126336  IMO5RT: AD:=5DD1; IF A><0 GO FAR EEILPAR
```

```
126341            IF D=0 THEN T:=RTREF=:D FI; AD=:5DD2
126346            X:=D; CALL GOODRT; GO FAR EEILPAR
126351            5DD2; CALL FSEMA; GO FAR IIM5RT
126354            CCPUDF=:B
126356            CALL 5MBREACTIVATE; GO FAR IIM5RT; GO FAR IIM5RT
126361    *)FILL
126373    IABLOG: IF 5D11><0 GO FAR EEILPAR
126376            IF X:=5D12=0 OR X=RTREF THEN
126403              IF BACKGROUND><0 OR 5SEMAPHORE="S500E" GO FAR ILLFUNC
126411            FI; IF X.ACTPRI BIT 17 GO FAR EEILPAR
126414            CALL GOODRT; GO FAR EEILPAR
126416            5DD1; CALL FSEMA; GO FAR 5ABPROG
126421            X.PSTAT\/110000; GO FAR INPRABORT
126424
126424    % CHECK FOR LEGAL RT-DESCRIPTION ADDRESS
126424    % ENTRY:      X=RT-DESCRIPTION ADDRESS
126424    % EXIT:       ERROR
126424    % EXIT+1:     OK, LEGAL RT-DESCRIPTION ADDRESS
126424    %
126424    GOODRT: IF X>>=SEGSTART OR X<RTSTART GO NGOOD
126432            A:=X-RTSTART; A=:D:=0; T:=5RTSIZE; *RDIV ST
126440            IF D><0 GO NGOOD; EXITA
126443    NGOOD:  EXIT
126444    *)FILL
126460    *)KILL YENDC; YENDC=*
126460    *7ENDC/
067470
067470    %=============================================================================
067470    %              F S E M A
067470    %
067470    % CHECK IF THE PROGRAM HAS RESERVED A ND-500 PROCESS
067470    % ENTRY:      D=RT-DESCRIPTION ADDRESS
067470    % EXIT:       THE PROGRAM HAS NOT RESERVED ANY ND-500 PROCESS
067470    % EXIT+1:     X=PROCESS DESCRIPTION OF THE RESERVED ND-500 PROCESS
067470    %
067470    FSEMA:  X:="BRESLINK"+D
067472            DO WHILE X:=X.RESLINK><D
067475               IF X>>="S500S+5SEMSIZE" AND X<<="S500E+5SEMSIZE" THEN EXITA FI
067504            OD; EXIT
067506    *)FILL
067511    *)KILL 7ENDC; 7ENDC=*
067511    *YENDC/
126460
126460    ILOGOFF:
126460    IPRABORT: IF 5D11><0 OR 5D12=0 OR A>>=CMXACTPROG GO FAR EEILPAR
126467            X:=ADRPROC(A); IF X.RTRES=0 GO FAR OKRET; A=:D
126475            IF 5FUNCTION=PRSTOP THEN
126501               IF D=RTREF THEN ABREL; GO CPRABORT FI
126506               CALL PCHSYSTEM; X.PSTAT BONE 5SYSABORT
126511            ELSE
126512               IF D=RTREF THEN XN5REL; GO CPRABORT FI
126517               X.PSTAT BONE SOFFLOGG
126521            FI; GO FAR INPRABORT
126522    *)FILL
126531
126531    CPRABORT: A=:5FUNCTION=:X; T:="PARANT"; *LBYT
126535              IF A BIT BM2SEGM THEN SG2500 ELSE SG1500 FI
126542              A=:5SEMAPHORE.CRSEGM=:C5REENT
126545              GO FAR NOPAR
126546
```

```
126546    TARGET: RTREF=:CCPUDF.SPREF
126551            FOR X:=1 TO CMXACTPROC-1 DO
126556                  X=:D:=ADRPROC(X)
126560                  IF X.RTRES><0 AND A><RTREF THEN
126565                     *IOF
126566                     X.PSTAT BONE 5SYSABORT BONE 5BRK=:X.PSTAT
126572                     X.CDTSEGN BONE 52ESCSET=:X.CDTSEGN
126575                     X.RTRES; *IRW MLEVB DX
126577                     "SYSABORT"; *IRW MLEVB DP
126601                     MLEV; *MST PIE; MST PID
126604                     CALL FAR XPION
126605                  FI; X:=D
126606            OD; GO FAR NOPAR
126611    *)FILL
126621
126621    FFPRACTIVE; ZPREG-1=:ZPREG
126624            IF 5D12=0 OR A>>=CMXACTPROC GO FAR EEILPAR
126631            X:=ADRPROC(A); IF X.RTRES=0 GO FAR EEILPAR
126636            IF X.MAGNO><5D11 GO FAR EEILPAR
126642            CALL MMBREACTIVATE; GO FAR EEILPAR; GO FAR OKRET
126645
126645    % CHECK IF PROCESS IN EXECUTION QUEUE AND ACTIVATE PROCESS
126645    % IF FOUND IN EXECUTION QUEUE
126645    %
126645    % MMBREACTIVATE MUST BE CALLED IN PIOF
126645    %
126645    % ENTRY:        X=PROCESS DESCRIPTION
126645    % EXIT:         PROCESS NOT IN EXECUTION QUEUE OR IN TIME QUEUE
126645    % EXIT+1:       THE PROCESS IS ACTIVATED
126645    %
126645    INTEGER FFPLREG,FFPBREG
126647    MMBREACTIVATE: A:=X.CPUDF:=:B=:FFPBREG
126652            GO MBREFELL
126653    SMBREACTIVATE: 0=:FFPBREG
126654    MBREFELL: *IOF
126655            A:=L=:FFPLREG; X=:D:=X.MESSBUFF
126661            T:=5MBBANK; *LDATX XN5ST
126663            IF A-I5TMQU=0 THEN                                  % IN TIME-QUEUE
126665               CALL FR5TMQ; CALL TER500; 0/\0                   % REMOVE FROM TIME-QUEUE
126670               A:=1; CALL SPITMQ; CALL ITO500XQ; GO FFPXX       % ACTIVATE PROCESS
126674            FI; X=:L:=MAILINK
126676            DO WHILE -1><X
126701               IF X=L THEN
126703                  *LDATX XN5ST                                  % A:=X.N5STATUS
126704                  IF A><STOPPED THEN
126707                     D.CDTSEGN BONE 55REP=:X.CDTSEGN; GO FFPOK
126714                  FI
126714                  CALL TER500; 0/\0; CALL OKMONICO
126717    FFPXX:        CALL LOWACT500
126720    FFPOK:        MIN FFPLREG; GO FFPFELL
126722               FI; T:=5MBBANK; *LDXTX XLNK2
126724            OD
126725    FFPFEL: FFPLREG=:L
126727             IF FFPBREG=0 GO FAR FPLRET
126732             A=:B;  EXIT
126734    *)FILL
126751    *)KILL YENDC; YENDC=*
126751    *YENDC)
067511
067511    FPLRET: *PION; POF; EXIT
```

```
067514
067514  FFRACTIVE: MLEV; *MCL PIE; POF                          %!!!!!!*POF
067517             GO FAR FFPRACTIVE
067520  XPION: *PION
067521         MLEV; *MCL PIE; POF; EXIT
067525
067525  INTEGER YYPL(0); *N500D+5D12
067526  IIM5RT:  A:=153100; GO 5ABFELL
067530  5ABPROG: A:=153105
067531  5ABFELL: A=:D; MLEV; *PON; MST PIE
067535           CALL ALTOFF
067536          "YYPL"; *EXR SD
067540          GO FAR OKRET
067541
067541  % ENTRY: X=PROCESS DESCRIPTION
067541  %        A=X.PSTAT
067541  INPRABORT: *IOF
067542         A BONE 5BRK=:X.PSTAT; X.CDTSEGN BONE 52ESCSET=:X.CDTSEGN
067547         X.RTRES; *IRW MLEVB DX
067551         "SYSABORT"; *IRW MLEVB DP
067553         MLEV; *MST PIE; MST PID; PION
067557         GO FAR OKRET
067560
067560  % SUBROUTINE TO CHECK FOR USER SYSTEM
067560  % ENTRY:      NONE
067560  % EXIT:       THE ACTUAL USER IS USER SYSTEM OR
067560  %             THE ACTUAL PROGRAM IS AN RT-PROGRAM
067560  %
067560  PCHSYSTEM: IF BACKGROUND><0 THEN
067562                MLEV; *PON; MST PIE
067565                IF 5PASSTYPE><2 GO EC25
067571                MLEV; *MCL PIE; POF
067574             FI; EXIT
067575
067575  IISTUSER: MLEV; *PON; MST PIE
067600         IF 5PASSTYPE=2 GO FAR OKRET
067604  EC25:  ENAUTORISED; GO FAR ERET
067606
067606  IWSYSP: "N500DF+SYSPAR"; CALL AATRANS; A=:LOGBADR
067611         T:=40; 5P1; CALL CMOVB
067614         GO FAR NOPAR
067615
067615  IRMESS: IF 5D11=0 THEN
067617             IF 5D12=-1 THEN                              % -1= OWN PROCES
067623                X:=5SEMAPHORE
067624             ELSE
067625                IF A>>=CMXACTPROC GO FAR EEILPAR
067630                A*6SEMSIZE+"S500S"=:X
067633             FI; X.MESSBUFFR
067634          ELSE
067635             CALL MESSREAD; GO FAR ERET
067637          FI; A=:D:=5MBBANK; CALL 5ATRANS; A=:LOGBADR
067643          T:=5MESSIZE SH 1; X:=5P2
067646  XOKRET: 0=:ZAREG; MIN ZPREG; 0/\0; GO FAR RFUNC
067652  IRSYSP: "N500DF+SYSPAR"; CALL AATRANS; A=:LOGBADR
067655         T:=40; X:=5P1; GO XOKRET
067660  *)FILL
067706
067706  % SUBROUTINE TO FIND THE ACTUAL "MON 60" COMMUNICATION BUFFER
067706  % AND TO SET UP A LOGICAL ADDRESS TO THIS BUFFER (WINOW ADDRESS).
```

```
067706    %
067706    INTEGER POINTER CLREG
067707    XGETBUFF: A:=L=:"CLREG"
067711           A:=5SEMAPHORE-"S500S"=:D:=0; T:=5SEMSIZE; *RDIV ST
067717           A SH 12+5BUSTART=:5BUADR; 5BUBANK=:BUADR
067724           DBUADR; CALL 5ATRANS; A=:LOGBADR
067727           GO CLREG
067730
067730    % SUBROUTINE TO MOVE DATA FROM THE USER AREA ON ALTERNATIVE PAGE INDEX TABLE
067730    % TO THE "MON 60" COMMUNICATION BUFFER ON PAGE INDEX TABLE ZERO (WINDOW ADDR)
067730    % ENTRY:      T=NUMBER OF BYTES TO MOVE
067730    %             A=SOURCE ADDRESS
067730    % RETURN:     IN PIOF AND MONITOR LEVEL DISABLED
067730    %
067730    CMOVB: T=:D:=A; MLEV; *PON; MST PIE
067735           T=:A:=D; D BONE 16; X:=LOGBADR; *BSET ONE; MOVB; JMP *
067744           MLEV; *MCL PIE; POF
067747           EXIT
067750    *)FILL
067752
067752    IOSLOAD: MLEV; *PON; MST PIE
067755             T:=200=:D; D BONE 16; 5P3; X:=LOGBADR
067762             *BSET ONE; MOVB; JMP *
067765             GO NOPAR
067766    IOCONNFI: MLEV; *PON; MST PIE
067771           T:=200=:D; D BONE 16; 5P1; X:=LOGBADR; *BSET ONE; MOVB; JMP *; BSET ZRO
070002           X=:T:=5P3; *BSET ONE; LDD ,X; BSET ZRO; COPY ST DX; STD ,X
070011           GO NOPAR; *)FILL
070013
070013    % COMMON POINT TO GO THROUGH BEFORE CALLING THE SYSTEM MONITOR
070013    NOPAR: MLEV; *PON; MST PIE
070016           CALL ALTOFF; "N500DF"=:B
070021           IF BACKGROUND><0 THEN RTREF.ACTSEG/\177400+C500SEG=:T ELSE T:=C500SEG FI
070032           CALL MMEXY; T=:5OLDSEG; CALL SVREINFO      % GET IN THE ND-500 DATA SEGMENT
070035           C5REENT; CALL MMREENT                      % GET IN TH ACTUAL SYSTEM MONITOR REENTRANT SEGMENT
070037           IF 5FUNCTION=N5RES OR A=MRESSPES OR A=SSTDOM OR 5SEMAPHORE="S500E" THEN
                    AD:=DBUADR; X:=5SEMAPHORE.MESSBUFF; T:=5MBBANK; *AAX ABUFA; STDTX
070063              O=:5SEMAPHORE.MAGNO
070065              IF BACKGROUND><0 AND 5BCHFLAG=0 AND 5TTIFIELD.TYPRING BIT 5TERM THEN
                       X.DFOPP.TDRADDR
070075              ELSE
                       A:=0
070077              FI; A=:5SEMAPHORE.OUTDF
070100              "RSEGSTART-2"=:L; A:=0; D:=0
070101              FOR X:="DASEGSTART" STEP 2 TO L DO AD=:X.DSO OD            % CLEAR ND-500 DATA SEGMENT
070103              T:=BNSYVARIABLES=:D; X:=DSSYVARIABLES; A:=B+"FSYVARIABLE"
070107              *MOVB; JMP *                                               % INITIALIZE THE ND-500 DATA SEGMENT
070115              X:=EXMONC; T:=MCKSIZE SH 1=:D; KEXMC; *MOVB; JMP *
070122              IF BACKGROUND=0 THEN
070124                    "RTPWORKA"=:DSSYVARIABLES.S25; T:=1
070133              ELSE
070135                    IF 5PASSTYPE = 0 THEN T:=0 ELSE T:=1 FI
070141              FI; IF T >< 0 THEN 5SEMAPHORE.CDTSEGN BONE SPROK=:X.CDTSEGN FI
070142           FI; GO NOPA1; *)FILL
070147    NOPA1: IF 5FUNCTION=WISON OR A=TSTFUNC THEN
070155              SBUFFR=:B; D:=0
070203              FOR X:="S500S+5SEMSIZE" STEP 5SEMSIZE TO "S500E" DO
070212                 X.S1=:S1; B+1; D+1
070215              OD; A:=D=:SBUFFER.S0; "N500DF"=:B
070221           FI
070225
070234
```

```
070234          IF 5SEMAPHORE="S500E" THEN 5FUNCTION=:5D52; 146=:5FUNCTION FI
070244          X:="S500DF"; A:="N500DF+ZPREG"; T:=55DFSIZE=;D
070250          *MOVB; JMP *            % COPY ND-500 DATAF. TO THE ND-500 DATA SEGMENT
070252  5(BREL: *IOF
070253          X:=RTREF; CALL BRELEASE; *ION
070256          IF 5SEMAPHORE><"S500E" THEN CALL XONESCAPE FI
070263          CALL FPT2ENTRY; GO 5PT2RET; GO SYMNLOAD      % GO TO SYSTEM MONITOR ON A SEGMENT
070266  *)FILL
070300
070300  % RETURN FROM REENTRAN SYSTEM MONITOR ON SEGMENT
070300  SYMNLOAD: ESNLOAD=:"S500DF".S3                       % ZAREG: SYS.MON NOT LOADED
070303  5PT2RET: "N500DF"=:B; *IOF
070306  RESAGAIN: X:=RTREF; CALL BRESERVE
070310          IF A<0 THEN
070311             CALL FREXQU; CALL TOWQU; CALL ANTIJAMMER
070314             "STUPR"; *IRW MLEVB DP
070316             MLEV; *MST PID; ION
070321             GO 5PT2RET
070322          FI; *ION
070323          GO FAR XRET5; *)FILL
070336
070336  RWFLAG: MLEV; *PON; MST PIE
070341          CALL ALTOFF; "N500DF"=:B
070344          IF 5D12=-1 THEN
070350             X:=5SEMAPHORE
070351          ELSE
070352             IF A*5SEMSIZE+"S500S">>="S500E" OR A.RTRES=0 GO FAR EEILPAR
070363          FI; X=:5D32; X.5SEG/\377=:T
070367          IF BACKGROUND><0 THEN RTREF.ACTSEG/\177400; T+A FI
070375          CALL MMEXY; T=:5OLDSEG
070377          CALL SVREINFO; C5REENT; CALL MMREENT          % GET IN USER'S ORIGINAL SEGMENTS
070402          IF 5FUNCTION=5RFLAG THEN                      % GET IN USER'S REENTRANT SEGMENT
070406             AD:="FF500".DS0=:5DD2
070411          ELSE
070412             IF 5D32><5SEMAPHORE THEN
070416                IF BACKGROUND><0 AND PASSTYPE=0 THEN ENAUTHORISED; GO FAR ERET FI
070424             FI; AD:=5DD2=:"FT500".DS0
070427          FI; 0=:ZAREG; MIN ZPREG; 0/\0; GO FAR XRET6
070433  *)FILL
070455  *)KILL 7ENDC; 7ENDC=*
070455  */ENDC/
126751
126751  % TABLE OF ENTRY POIONTS TO GO TO AFTER RETURNING FROM THE SYSTEM MONITOR
126751  % THE "MON 60" FUNCTION CODE IS INDEX IN THIS TABLE.
126751  @ICR
126751  INTEGER ARRAY FUNCS:=(
126751          FRREG,FWREG,FPMREAD,FDMREAD,FPMWRITE,FDMWRITE,FSEGLOAD,FPLSWAPPER,
126761          FRRGS,FWRGS,FPRSTART,FCONNFI,FCLFIE,FRES5,FREL5,FFLIOPFI,
127001          FTIMUSED,FWISON,FERRSET,FRCNT,FWCNT,FMICPSTART,FDMEXA,FDMDEP,
127011          FPMEXA,FPMDEP,FDAMR,FDAMW,FMICSTOP,FFMCLEAR,FALLPSEG,FCSLOAD,
127021          5FMEMDEF,FRSTATU,FABRELS,FSRES,FSREL,FSCPLOOP,FDEFSWAP,FDELSWAP,
127031          FTSTFUNC,FRIREG,FG1VP,FTAKP,FSTSWAP,FSPLACE,FEPLACE,FRMICV,
127041          FLIMEM,FMRESSPES,RET5,RET5,RET5,RET5,RET5,FSPRTE,
127051          FGPRTE,FSSGTE,FGSGTE,FRPHSG,FSPRNM,FTSTUSER,FTOSWP,RET5,
127061          F5RFLAG,F5WFLAG,FFORGET,FRSYSP,FWSYSP,FS1PRIOR,FLINKTO,0,
127071          FWPHSG,FSTAPRLOG,FSTOLOG,FPRILOG,FRELLOG,FSTLAPR,FREL5,FPRABORT,
127101          FSOUTFIL,FRSWPDATA,F5LOGOFF,FMRELSPES,FSTAMLOG,FPRIMLOG,FSTOMLOG,FDFSYDOM,
127111          FSFSYDOM,FDLSYDOM,FLISYDOM,FLI5EXQ,FPLDEB,FABLOG,FPRACTIVE,0,
127121          FRESPLACE,FS5SBLSIZE,RET5,RET5,RET5,FSSTDOM,RET5,FREL3,
                RET5);
```

```
127122    ;R;
127122
127122    % SUBROUTINE TO REMOVE A MESSAGE FROM THE EXECUTION QUEUE AND FROM THE
127122    % TIME QUEUE
127122    % ENTRY:      B=ND-500 CPU DATAFIELD
127122    %             X=ND-500 MESSAGE
127122    %
127122    INTEGER POINTER FRQRET
127123    INTEGER FRQFLG
127124
127124    FRQUES: A:=L=:"FRQRET"; 0=:FRQFLG
127127            CALL CHEXQ; GO FRQU1                                    % PROC. IN EXEC. QUEUE?
127131            CALL TER500; 0/\0; MIN FRQFLG; CALL IFM500XQ            % YES, REMOVE MESSAGE
127135    FRQU1:  CALL FR5TMQ; T:=5MBBANK; A:=MFREE; *STATX XN5ST         % REMOVE MESSAGE FROM ND-500 TIME QUEUE
127141            IF FRQFLG><0 THEN CALL LOWACT500 FI                     % IF ND-500 IS STOPPED, THEN RESTART IT
127144            GO FRQRET
127145    *)FILL
127153    *)KILL VENDC; VENDC=*
127153    *)ENDC)
070455
070455
070455    XRET5:  X:="ZPREG"+B; T:=55DFSIZE=:D; "S500DF" % COPY FROM THE "SAVED ND-500 DATAF." ON
070462            *MOVB; JMP *                          % THE ND-500 DATA SEGMENT TO THE ND-500 DATAF. (MONITOR CALL WORKI
070464    XRET6:  5SEMAPHORE.CPUDF=:CCPUDF
070467            CALL USVREINFO; T:=50LDSEG; CALL MMEXY     % GET USER'S ORIGINAL SEGMENTS
070472            IF 5FUNCTION=SSTDOM AND 5SEMAPHORE.PSTAT NBIT 5BRK THEN  % RETURN TO S3 COM.PROC
070502               CALL XOFFESCAPE; CALL RELREFS; CALL ALLRELEASE
070505               T:=5TTIFIELD; *IOF
070507               T=:CTTIFIELD
070510               X:=RTREF; 5SEMAPHORE=:B; CALL BRELEASE
070514               "N500DF"=:B; X:=CTTIFIELD; CALL CHDFPAGE; GO FAR IRET5
070521            FI
070521            IF ZAREG><0 THEN
070523               IF A=ESCSTOP GO FAR FROMESC
070526               IF 5FUNCTION><N5REL AND A><N5RES AND A><XN5REL AND A><ABREL GO FAR RET5
070543            FI; *PIOF
070544            FUNCS(5FUNCTION); *PION
070547            A=:P                        % GO TO ROUTINE ACCORDING TO "MON 60" FUNCTION CODE
070550
070550    FRREG:  X:=5P2; AD:=5DD2; GO FAR STDS0
070553    FLIMEM: X:=5P1; T:=70; GO FAR RFUNC
070556
070556    FRSWPDATA:
070556    FPMREAD:
070556    FDMREAD:
070556    FDAMR:  X:=5P4; AD:=5DD4; CALL 5ALTON; AD=:X.DS0; *BSET ZRO
070563            X:=5P3; T:=5D42; GO FAR RFUNC
070566
070566    FRRGS:  T:=NREGS SHZ 2; X:=5P1; GO FAR RFUNC
070572
070572    FPRSTART: A:=5SEMAPHORE; CALL FAR X5DCN
070574            X:=5P1; AD:=5DD1; CALL 5ALTON; AD=:X.DS0; *BSET ZRO
070601            X:=5P2; T:=200; GO FAR RFUNC
070604
070604    FCONNFI: X:=5P5; AD:=5DD5; GO FAR STDS0; *)FILL
070635
070635    % SUBROUTINE TO
070635    %       1. RELEASE THE ND-500 CPU FROM SPECIAL USE IF RESERVED FOR SPECIAL USE
070635    %          BY THE CALLING PROGRAM.
070635    %       2. RELEASE ALL LOGGING FACILITIES RESERVED BY THE CALLING PROGRAM
```

```
070635    %
070635    INTEGER RGL
070636    RELREFS:
070636            IF 5SEMAPHORE.CPUDF.SPREF=RTREF THEN 0=:X.SPREF; X.INITFLAG BZERO BRESPLACE=:X.INITFLAG FI
070650            IF 5HRTP=T THEN
070653               *PIOF
070654               A:=L=:RGL
070656               0=:5HRTP=:5HIFLAG; CALL RHIFEXQ; "N500DF"=:B
070663               RGL=:L; *PION
070666            FI; IF 5MLOG=RTREF THEN 0=:5MLOG=:5MLOPROC FI
070674            EXIT
070675    *)FILL
070704
070704    INTEGER CCBSTATE=?
070704    FREL3: A:=5SEMAPHORE; CALL FAR X5DCN; GO FABREL
070707    FABLOG: A:=5SEMAPHORE; CALL RELREFS; CALL FAR X5DCN; GO IFXN5REL
070713    FREL5: A:=5SEMAPHORE; CALL FAR X5DCN
070715            IF 5SEMAPHORE.PSTAT BIT 5BRK AND X.5RUNSTATUS><USRBRK THEN
070725               IF A=5IDLE GO FAR RET5
070730               5IDLE=:X.5RUNSTATUS; GO FPRSTART; *)FILL
070736            FI
070736    FABREL:
070736    AXFREL5:
070736            CALL XOFFESCAPE; CALL ALLRELEASE
070740            IF BACKGROUND><0 THEN T:=5TTIFIELD ELSE T:=0 FI
070745            X:=RTREF; 5SEMAPHORE=:B; MLEV; *MCL PIE
070752            T=:CTTIFIELD; CALL BRELEASE
070754            "N500DF"=:B; CALL RELREFS
070757            IF X:=CTTIFIELD><0 THEN
070761               *IOF
070762               CALL CHDFPAGE; X.FLAGB BZERO 5ESC2SET BONE 5ESCON=:X.FLAGB
070767               *ION
070770            FI
070770            IF 5FUNCTION=XN5REL OR A=ABREL GO FXN5REL
070777            IF 5SEMAPHORE.PSTAT BIT 5BRK THEN
071003               0=:RTREF.RSEGM
071005               IF 5SEMAPHORE.5ERRFLAG=1 THEN
071012                  X:=RTREF; CALL BRELEASE; X:=5SEMAPHORE
071015                  X."LRET"=:L; X.5BRG=:B; MLEV; *MST PIE
071023                  EXIT
071024               FI
071024               IF X.PSTAT BIT 5SYSABORT AND A BIT SOFFLOGG THEN
071031    FXN5REL:      *PIOF
071032                  T:=X.MESSBUFF; 5SEMAPHORE.CPUDF=:B; X:=T; CALL FAR FRQUES
071040                  "N500DF"=:B; *PION
071043                  IF 5SEMAPHORE.PSTAT BIT 5SYSABORT AND A BIT SOFFLOGG THEN
071051    IFXN5REL:        *IOF
071052                     5SEMAPHORE.CDTSEGN/\1400=:T; RTREF.STATUS/\40377\/T=:X.STATUS
071063                     IF 5FUNCTION=ABLOG AND 5SEMAPHORE.PDCFLG><0 THEN      % SINTRAN "SYSTEM DETECTED ERROR"
071072                        X=:B:=RTREF; CALL BRELEASE                         % RELEASE PROC.DESC.
071075                        "N500DF"=:B; 5SEMAPHORE.PDCFLG
071101                        GO FXN2REL
071102                     FI; X:=RTREF; CALL FRWQU; CALL FTIMQU
071105                     0=:X.RSEGM; GO ABRETXIT
071107                  FI; T:=5BUSER; GO FXNREL; *)FILL
071131            FI
071131            IF A BIT 5SYSABORT OR A BIT SOFFLOGG THEN
071135               T:=5BUSER
071136            ELSE
071137               IF 5FUNCTION=N5S3REL THEN T:=5ND5ESC ELSE T:="5BUSER+1" FI
```

```
071146                  FI
071146      F.NREL:     IF BACKGROUND=0 GO FAR RET5
071151                  *IOF
071152                  T=:CCBSTA; X:=CTTIFIELD; CALL CHDFPAGE; CCBSTA=:X.BSTATE
071157                  "YBRTWT"
071160      F.NZREL:    *IRW MLEVB DP
071161                  X:=RTREF; CALL BRELEASE                 % RELEASE N500DF DATAFIELD
071163                  MLEV; *MST PID; MST PIE; ION; JMP *
071170              FI; MLEV; *MST PIE
071172              GO RET5
071173      *)FILL
071202      INTEGER CCBSTATE
071203
071203      FRSYSP: T:=20; X:=5P1; GO FELL2
071206      FRSWAPA: T:=30; X:=5P1; GO RFUNC
071211      FRPROC: T:=5PRSELSIZE; GO FELL1
071213      FRSGTE: T:=PHSELSIZE; X:=5P2; GO FELL2
071216      FRSGTE: T:=PHSELSIZE; GO FELL1
071220      FRPHSG: X:=5P5; AD:=5DD5; CALL 5ALTON; AD=:X.DS0; *BSET ZRO
071225              T:=5D32; X:=5P4; GO RFUNC
071230      FRPRTE: T:=PRSELSIZE
071231      FELL1:  X:=5P2
071232      FELL2:  T SH 1; GO RFUNC
071234
071234
071234      FRCNTS: X:=5P3; T:=5D22 SH 1; GO RFUNC
071240      FRES5: X:=5P2; T:=11; GO RFUNC
071243      FRSTATU: X:=5P1; AD:=5DD1; CALL 5ALTON; AD=:X.DS0; *BSET ZRO
071250              X:=5P2; AD:=5DD2; *BSET ONE
071253              AD=:X.DS0; *BSET ZRO
071255              AD:=5DD3; X:=5P3; *BSET ONE
071260              AD=:X.DS0; GO RET5
071262
071262      FRIREG:
071262      FRMICV: X:=5P1; AD:=5DD1; GO STDS0
071265      FSRFLAG:
071265      FSFSYDOM:
071265      FSPRTE:
071265      FDMEXA:
071265      FPMEXA: X:=5P2; AD:=5DD2
071267
071267      STDS0: CALL 5ALTON; AD=:X.DS0; GO RET5
071272      FSSTDOM:
071272              CALL FAR X5DCN; GO RET5         % START STANDARD DOMAIN FROM ND-100
071274      *)FILL                                  % DISCONNECT FROM XMSG
071277
071277      FFORGET;
071277              1=:5MSINIT                      % STOP-ND-500
071301              0=:RELBFLG                      % INDICATE BUFFERS NOT LONGER ALLOCATED
071302      FMRELSPES: XN5REL=:5FUNCTION; GO FAR XXFREL5    % CLEAR "BUFFER-RELEASE-GOING-ON" FLAG
071305
071305      % COPY DATA FROM THE "MON 60" COMMUNICATION BUFFER TO THE USER AREA
071305      % ON ALTERNATIVE PAGE INDEX TABLE
071305      % ENTRY:        T=NUMBER OF BYTES TO COPY
071305      %               X=DESTINATION ADDRESS IN USER AREA
071305      RFUNC: T=:D BONE 16; LOGBADR; CALL 5ALTON; *MOVB; JMP *
071313
071313      @ICR
071313      FWRGS:; FPMWRITE:; FDMWRITE:; FPSLOAD:; FDSLOAD:; FWRGS:; FCLFIE:; FMICSTOP:;
071313      FFLIOPFI:; FTIMUSED:; FERRFSET:; FWCNTS:; FDMDEP:; FPMDEP:;
```

```
071313    FDAMW:; FFMCLEAR:; FERRSET:; FMICPSTART:; FALLPSEG:; FALLDSEG:; FDEFSWAP:;
071313    FDELSWAP:; FSRES:; FSREL:; FTSTFUNC:; FGIVP:; FTAKP:; FSTSWAP:; FSEGLOAD:;
071313    FPLSWAPPER:; 5FMEMDEF:; FWISON:; F5WFLAG:; FS1PRIOR:;
071313    FSPRNM:; FTOSWP:; FWSYSP:; FEPLACE:; FSPLACE:; FSCPLOOP:; FLINKTO:;
071313    FMICBRK:; FWPHSG:; FSTAPRLOG:; FSTOLOG:; FPRILOG:; FRELLOG:; FSTLAPR:;
071313    FISTUSER:; FPRABORT:; FSOUTFIL:; F5LOGOFF:; FMRESSPES:;
071313    FCSLOAD:; FSTAMLOG:; FPRIMLOG:; FSTOMLOG:; FDFSYDOM:; FDLSYDOM:;
071313    FLI5EXQ:; FLISYDOM:; FPLDEB:; FRESPLACE:; FS5SBLSIZE:;
071313    WOR;
071313
071313    RET5:
071313    RELBUF: CALL ALTOFF
071314    MIRET:
071314            IF BACKGROUND><0 THEN 5TTIFIELD FI
071317            *IOF
071320            A=:CTTIFIELD; X:=RTREF
071322            IF 5SEMAPHORE="S500E" THEN A=:B; CALL BRELEASE; "N500D"=:B FI
071332            IF BACKGROUND><0 THEN
071334               X.STATUS BZERO 11=:X.STATUS
071337               X:=CTTIFIELD; CALL CHDFPAGE; 5BUSER=:X.BSTATE
071343    IRET5:     X.FLAGB BZERO 5ESC2SET=:X.FLAGB
071346            ELSE
071347               5SEMAPHORE.CDTSEGN/\1400=:T; RTREF.STATUS/\176377\/T=:X.STATUS
071360            FI: 5SEMAPHORE.PSTAT BZERO 5BRK=:X.PSTAT
071364            GO MXRET
071365
071365    OKRET: MLEV; *PON; MST PIE; BSET ZRO
071371            0=:"N500DF".ZAREG; B:=X; MIN ZPREG; 0/\0; GO RET5
071377    *)FILL
071416
071416    %       S V R E I N F O  -  U S V R E I N F O
071416    %
071416    % SAVE/UNSAVE REENTRANT SEGMENT INFORMATION FOR THE USER PROGRAM.
071416    %
071416
071416    INTEGER CCBB,CCLL
071420    SVREINFO: K:="0"; RTREF.RSEGM=:5RSEGM; GO SVUSV
071425    USVREINFO: K:=1; 5RSEGM=:RTREF.RSEGM
071431    SVUSV: MLEV; *MCL PIE
071433           A:=L=:CCLL
071435           A:=B+"5SBITMAP-BXBITMAP":=:B=:CCBB; X:=RTREF
071442           IF K THEN CALL BMTRTD ELSE CALL BMFRTD FI
071447           CCBB=:B
071451           CCLL=:L
071453           MLEV; *MST PIE
071455           EXIT
071456
071456
071456    %       5 A T R A N S  -  A A T R A N S
071456    %
071456    % SUBROUTINE TO SET WINDOW ADDRESS TO BUFFER ADDRESS
071456    %
071456    % ENTRY FOR A A T R A N S; A= ABSOLUTE ADDRESS OF BUFFER
071456    % ENTRY FOR 5 A T R A N S; AD=ABSOLUTE ADDRESS OF BUFFER
071456    % (5PATRANS AND PAATRANS IS CALLED IN PIOF)
071456    %
071456    % EXIT:  A= LOGICAL ADDRESS OF BUFFER
071456    %
071456           INTEGER SVT,SVA,SVD,SVX; TRIPLE SVTAD=SVT
071462           DOUBLE POINTER BGP:=177000+5BFPAGE+5BFPAGE
```

```
071463          INTEGER MU5BFPAGE(0); *5BFPA@12
071464    PAATRANS: A=:D:=0
071466    5PAATRANS: K:=1; GO ATRA1
071470    AATRANS: A=:D:=0
071472    5ATRANS: K:="0"; *PIOF
071474    ATRA1: TAD=:SVTAD; X=:SVX; AD SHZ -12; T:=D SH 10
071501           A:=RTREF.WINDOW/\377\/T=:X.WINDOW
071506           162000; AD=:BGP
071510           1777/\SVD+MU5BFPAGE=:SVA
071514           TAD:=SVTAD; X:=SVX
071516           IF K THEN EXIT FI
071521           *PION
071522           EXIT
071523    *FILL
071532
071532
071532    %       5 S G A L T O N
071532    %
071532    % SET THE PAGE TABLE OF CURRENT SEGMENT AS CURRENT ALTERNATIVE PAGE TABLE
071532    %
071532    5SGALTON: *PIOF
071533           TAD=:SVTAD; X=:SVX
071535           RTREF.ACTSEG/\377*5SEGSIZE+SEGSTART
071542           A.LOGADR SH 1/\600; GO INALT
071547
071547
071547    %       5 A L T O N
071547    %
071547    % SUBROUTINE TO SET THE CALLING PROGRAMS ALTERNATIVE PAGE TABLE AS
071547    % CURRENT ALTERNATIVE PAGE TABLE
071547    %
071547    5ALTON: *PIOF
071550           TAD=:SVTAD; X=:SVX
071552           "N500DF".OLDPAGE/\600
071555    INALT: A=:T; CURPROG.ACTPRI/\177177\/T=:X.ACTPRI/\3773
071564           *TRR PCR; BSET ONE
071566           TAD:=SVTAD; X:=SVX; *PION
071571           EXIT
071572
071572    %       T S 3 C O M P R O C
071572    %
071572    % CALL ROUTINES IN SINTRAN III OP.COM
071572    %
071572    TS3COMPROC: T:=3; GO IFSYINT
071574
071574
071574    %       F I L S Y S I N T E R F A C E
071574    %
071574    % CALL ROUTINES IN THE FILESYSTEM
071574
071574    INTEGER POINTER OFLDX:=OFLCK
071575    FSYSINTERFACE: T:=6
071576    IFSYINT: A:=L=:5LREG; "S500DF-ZPREG".5SEMAPHORE=:B
071603           OFLDX; CALL XLOCK; IF 6=T THEN "FRSG1" ELSE A:=0 FI
071613           A=:RTREF.RSEGM; X.ACTSEG/\177400; T+A; CALL MMEXY
071621           TAD:=5TADRG; X:=5XRG; 0=:5ERRFLAG; CALL FSYSENTRY; MIN 5ERRFLAG
071626           TAD=:5TADRG; X=:5XRG
071630           5SEG/\377=:T; RTREF.ACTSEG/\177400; T+A; CRSEGM=:X.RSEGM; CALL MMEXY
071642           IF CDTSEGN NBIT OFLDUNLOCK THEN OFLDX; CALL XUNLOCK FI
071647           5LREG=:L; IF 5ERRFLAG=0 THEN L+1 FI
```

```
071654          TAD:=5TADRG; A=:5AREG; X:=5XRG; DBASE=:B; 5AREG; EXIT
071663
071663  *)FILL
071703
071703  %       S E G M O N C
071703  %
071703  % EXECUTES MONITOR CALLS NEEDING A SEGMENT TO TRANSFER DATA TO/FROM
071703  % THE MONITOR CALL REQUIRE STANDARD PARAMETER LISTE
071703  %
071703  % ENTRY:       X=CURRENT PROCESS DESCRIPTION
071703  %              T=SEGMENT NUMBER TO TRANSFER DATA TO/FROM
071703  %              THE PARAMETERS ARE FOUND IN THE PROCESS DESCRIPTION
071703  %
071703
071703          INTEGER CMONINSTR(0); *MON
071704  SEGMONC: X=:B:=L=:5LREG; RTREF.ACTSEG/\177400; T+A
071713          L.S0+CMONINSTR; 0=:RTREF.RSEGM; CALL MMEXY; A=:D
071722          CALL FAR 5SGALTON
071723          A:=B+"PLIST"; *BSET ZRO SSK; EXR SD; BSET ONE SSK; BSET ZRO
071731          A=:5ARG; CRSEGM=:X.RSEGM
071734          GO RSGM1
071735
071735  %       R S G M O N C
071735  %
071735  % EXECUTE MONITOR CALLS NEEDING A SEGMENT TO TRANSFER DATA TO/FROM
071735  % THE PARAMETER TO THE MONITOR CALL SHOULD BE IN THE REGISTERS
071735  %
071735  % ENTRY:       THE PARAMETERS ARE FOUND IN THE PROCESS DESCRIPTION (5TADRG+5XRG)
071735  %              X=CURRENT PROCESS DESCRIPTION
071735  %              T=SEGMENT NUMBER TO TRANSFER DATA TO/FROM
071735  % EXIT:        NOT SKIP-RETURN FROM THE MONITOR CALL
071735  %              REGISTERS IN PROCESS DESCRIPTION
071735  % EXIT+1:      SKIP-RETURN FROM THE MONITOR CALL
071735  %              REGISTERS IN PROCESS DESCRIPTION
071735  %
071735  RSGMONC: X=:B:=L=:5LREG; RTREF.ACTSEG/\177400; T+A
071744          L.S0+CMONINSTR; 0=:RTREF.RSEGM; CALL MMEXY; T=:5CURSEGS
071753          CALL FAR 5SGALTON; *BSET ZRO
071755          A=:L; TAD:=5TADRG; X:=5XRG
071760          *BSET ZRO SSK; BSET ONE; EXR SL; BSET ONE SSK; BSET ZRO
071765          TAD=:5TADRG; X=:5XRG; CRSEGM=:RTREF.RSEGM; T:=5CURSEG
071773  RSGM1:  X:=2; *BSKP ZRO SSK; SAX 1
071776          CALL ALTOFF; CALL MMEXY
072000          5ARG; T:=DBASE=:B:=5LREG+X=:P
072006  *)FILL
072016
072016
072016  %       5 G E T L
072016  %
072016  % GET ONE LOCATIOIN FROM A SEGMENT
072016  %
072016  % ENTRY:       X=ADDRESS
072016  %              T=SEGMENT
072016  %
072016  % EXIT:        ERROR
072016  %
072016  % EXIT+1:      A=VALUE OF LOCATION
072016  %
072016  5GETL:  A:=L=:5LREG
072020          X=:L; RTREF.RSEGM=:D; 0=:X.RSEGM; X:=L
```

```
072026          CALL MMEXY; CALL FAR 5SGALTON; X.S0; CALL ALTOFF; A:=:D; X=:B
072034  INGETL: A=:RTREF.RSEGM; CALL MMEXY
072037          5LREG=:L; X:=DBASE; X:=:B; A:=D; EXITA
072045
072045  %       5 P U T L
072045  %
072045  % INSERT VALUE IN ONE LOCATION OF A SEGMENT
072045  %
072045  % ENTRY:        A=VALUE
072045  %               X=ADDRESS IN SEGMENT
072045  %               T=SEGMENT NUMBER
072045  %
072045  % EXIT:         ERROR
072045  %
072045  % EXIT+1:       OK
072045  %
072045  5PUTL:  X:=:L; X=:5LREG; A=:B; RTREF.RSEGM=:D; 0=:X.RSEGM
072054          X:=L; CALL MMEXY; CALL FAR 5SGALTON; A:=B; A=:X.S0; CALL ALTOFF
072062          X=:L; A:=D=:RTREF.RSEGM; X:=L
072067          CALL MMEXY; 5LREG=:L; DBASE; A:=:B; EXITA
072075  *FILL
072103
072103  %       G E T L S
072103  %
072103  % SUBROUTINE TO GET FILE INFORMATION FROM THE DIRECTORY TABLE AND
072103  % THE NAME TABLE
072103  %
072103  % ENTRY:        X=ADDRESS OF DUNIT IN DIRECTORY TABLE
072103  %               A=ADDRESS OF PROCESS DESCRIPTION
072103  %               T=FILE SEGMENT NUMBER
072103  %
072103  % EXIT:         ERROR
072103  %
072103  % EXIT+1:       THE FILE INFORMATION IS IN THE PROCESS DESCRIPTION
072103  %
072103  GETLS:  A=:B:=L=:5LREG
072106          X=:L; RTREF.RSEGM=:D; 0=:X.RSEGM; X:=L; CALL MMEXY; CALL FAR 5SGALTON
072116          X.S0; *BSET ZRO
072120          A=:5DUNIT;=X+"LUNIT-DUNIT"=:X; *BSET ONE
072125          X.S0; *BSET ZRO
072127          A=:5LUNIT/\377*NTLEN+"NAMTA+NFLAG"=:X; *BSET ONE
072135          X.S0; *BSET ZRO
072137          A=:6NFLAG:=X+"SECTO-NFLAG"=:X; *BSET ONE
072144          X.S0; *BSET ZRO
072146          A=:5SECTO:=X+"PAVA1-SECTO"=:X; *BSET ONE
072153          X.S0; *BSET ZRO
072155          A=:1PAVA; *BSET ONE
072157          X.S1; *BSET ZRO
072161          A=:2PAVA; CALL ALTOFF; A:=D; GO FAR INGETL
072165  *FILL
072201
072201  %       5 G N S E G   -   5 G S Y D S G
072201  %
072201  % REMOVE THE DATA SEGMENT, GET THE NAME SEGMENT/STANDARD DOMAIN SEGMENT
072201  % SET5 PT2 AS ALTERNATIVE PAGE TABLE
072201  %
072201  % ENTRY:        X=D= ADDRESS OF PROCESS DESCRIPTION
072201  %               T= LOGICAL ADDRESS OF CURRENT "DATA BUFFER"
072201  %
```

```
072201    % EXIT:       A= ADDRESS OF 5GDSEG
072201    %             T= UNCHANGED
072201    %
072201    5GSYDSG: A:=L=:X."LRET"; T=:X.5TRG
072204             RTREF.ACTSEG/\177400+20; GO I5GNSEG
072211    5GNSEG:  A:=L=:X."LRET"; T=:X.5TRG
072214             RTREF.ACTSEG/\177400+"N500DF".NNAMSEG
072221    I5GNSEG: A=:T; CALL MMEXY
072223             *PIOF
072224             RTREF.ACTPRI/\177177+400=:X.ACTPRI/\3773; *TRR PCR
072233             *PION; BSET ZRO
072235             A:="5GDSEG"; T:=D.5TRG; X:=X."LRET"=:P
072242
072242    %       5 G D S E G
072242    %
072242    % REMOVE THE NAME SEGMENT/STANDARD DOMAIN SEGMENT, GET THE DATA SEGMENT
072242    % RESET THE ALTERNATIVE PAGE TABLE
072242    %
072242    % ENTRY:      X=D= ADDRESS OF PROCESS DESCRIPTION
072242    %
072242    % EXIT:       NONE
072242    %
072242    5GDSEG:  *BSET ZRO
072243             A:=L=:X."LRET"
072245             X.5SEG/\377=:T; RTREF.ACTSEG/\177400; T+A; CALL MMEXY
072255             *PIOF
072256             X.ACTPRI/\177177=:X.ACTPRI/\377; *TRR PCR
072263             *PION
072264             D."LRET"=:P
072267
072267    *FILL
072300
072300    INTEGER ARRAY XSEMS:=(5NAMSEM,CSSEM,PLSSEM,FIXSEM,CSSEM,PLSSEM,SYDSEG,SWORKA)
072310    INTEGER 5NAMSEM(0)
072310    *       0;0;*-2;2
072314    INTEGER CSSEM(0)
072314    *       0;0;*-2;2
072320    INTEGER PLSSEM(0)
072320    *       0;0;*-2;2
072324    INTEGER FIXSEM(0)
072324    *       0;0;*-2;2
072330    INTEGER SYDSEG(0)
072330    *       0;0;*-2;2
072334    INTEGER SWORKA(0)
072334    *       0;0;*-2;2
072340
072340    %       R E S N A M S E G
072340    % SUBROUTINE TO RESERVE SOME RESOURCES
072340    %
072340    % ENTRY:      T=0 : NAME SEGMENT
072340    %            T=1 : LOAD CONTROL STORE FUNCTION
072340    %            T=2 : PLACE SWAPPER FUNCTION
072340    %            T=3 : WAIT FOR FIX-PAGES
072340    %            T=4 : RESERVE CONTROL STORE
072340    %                  RETURN:      CONTROL STORE ALREADY RESERVED
072340    %                  SKIPRETURN: CONTROL STORE IS RESERVED BY CALLER
072340    %            T=5 : RESERVE "PLACE-SWAPPER"
072340    %                  RETURN:      "PLACE-SWAPPER" ALREADY RESERVED
072340    %                  SKIPRETURN: "PLACE-SWAPPER" IS RESERVED BY CALLER
072340    %            T=6: RESERVE THE SYSTEM-DOMAIN SEGMENT
```

```
072341    %              T=7: RESERVE RTPWORKA FOR RT-PROGRAMS
072343    %
072342    INTEGER RXRG=?,RTRG=?,RLRG=?,RBRG=?
072344    RESNAMSEG: A:=L=:5LREG
072342    RESN1: MLEV; *MCL PIE
072344           X=:RXRG; T=:RTRG
072346           XSEMS(T)=:B
072351           X:=RTREF; CALL BRESERVE
072353           IF A<0 THEN
072354              IF 4>RTRG OR 6<=T THEN
072363                 CALL FREXQU; CALL TOWQU; CALL ANTIJAMMER
072366                 "STUPR"; *IRW MLEVB DP
072370                 MLEV; X:=RXRG; T:=RTRG; *MST PID; MST PIE
072375                 GO RESN1
072376              ELSE
072377                 X:=RXRG; MLEV; *MST PIE
072402                 5LREG=:L; GO RESRET
072405              FI
072405           FI; T:=RTRG; X:=RXRG
072407    RESN2: MLEV; *MST PIE
072411           5LREG+1=:L
072414    RESRET: DBASE=:B; EXIT
072417
072417    %       R E L N A M S E G
072417    % RELEASE SOME RESOURCES
072417    %
072417    % ENTRY:       SAME AS FOR RESNAMSEG
072417    %
072417    RELNAMSEG: A:=L=:5LREG
072421           MLEV; *MCL PIE
072423           X=:RXRG; XSEMS(T)=:B
072427           IF X:=RTRES=RTREF THEN CALL BRELEASE FI
072434           X:=RXRG; GO RESN2
072436
072436    *)FILL
072450
072450    INTEGER RXRG,RTRG,RLRG,RBRG
072454
072454    %       A L L R E L E A S E
072454    %
072454    % RELEASE ALL ND-500 SEMAPHORES
072454    %
072454    ALLRELEASE: MLEV; *MCL PIE
072456           A:=L=:RLRG; X=:RXRG:=B=:RBRG
072463           FOR X:=0 TO 7 DO
072467              X=:RTRG:=XSEMS(X)
072471              IF X.RTRES=RTREF THEN X=:B:=A; CALL BRELEASE FI
072500              X:=RTRG
072501           OD
072503    ALLR1: RTREF.BRESLINK
072505           DO WHILE A><RTREF
072510              IF A>="9SFIS" AND A<"9EFIS" THEN
072516    ALLR2:       X:=RTREF; A=:B; CALL BRELEASE; GO ALLR1
072522              ELSE
072523                 IF A>="DEVBU" AND A<"ENDBU" GO ALLR2
072531                 IF T:="MXSIBAS"><0 THEN
072534                    IF A>="DSIO" AND A<"EDSIBAS" GO ALLR2
072542                 FI
072542              FI; A.RESLINK
072544           OD; T:=RLRG; X:=RBRG=:B:=RXRG; MLEV; *MST PIE
```

```
072553          T=:P
072554
072554  *)FILL
072566
072566
072566
072566  %       A C T D R I V E R
072566  %
072566  % ACTIVATE THE N500 DRIVER ON MONITOR LEVEL
072566  %
072566  % ENTRY:      A=0 MEANS STOP PROCESS; X=PROCESS DESCRIPTION ADDRESS
072566  %             A=ADDRESS OF PROCESS DESCRIPTION
072566  %
072566
072566  INTEGER CXX
072567  ACTDRIVER: *PIOF
072570          IF A=0 THEN
072571             X=:CXX
072572             0=:X.LPRCOUNT
072573             A:=L=:X."LRET":=X.CPUDF=:B
072577             X:=X.MESSBUFF; CALL FAR FRQUES
072601             CXX."LRET"=:L
072604             *PION
072605             DBASE=:B; EXIT
072610          FI; *IRW MLEVB DX
072611          "P500C"; *IRW MLEVB DP
072613          MLEV; *MST PIE; MST PID; PION
072617          EXIT
072620  *)FILL
072623
072623  %       P 5 0 0 C
072623  %       MONITOR LEVEL
072623
072623  P500C: *POF
072624          GO N500C
072625
072625  %       S E T I O W A I T
072625  %
072625  % SETS THE CALLING PROGRAM IN I/O WAIT
072625  %
072625  % ENTRY:      X=ADDRESS OF CURRENT PROCESS DESCRIPTION
072625  %
072625
072625  SETIOWAIT; *IOF
072626          IF X.F5BUFF=0 THEN
072630             X.PSTAT BONE 5REWA=:X.PSTAT
072633             RTREF.STATUS BONE 5WAIT=:X.STATUS
072637             "RWAIT"; *IRW MLEVB DP
072641             MLEV; *MST PIE; MST PID
072644          FI; *ION
072645          EXIT
072646
072646  *)FILL
072651
072651
072651  %       C H 5 M X T I M E
072651  %
072651  % SUBROUTINE TO THE TIMER RT-PROGRAM
072651  % ABORT BATCH-JOBS USING NORD 500 WHEN THE N500 CPU TIME
072651  % IS GREATER OR EQUAL TO THE MAX CPU TIME IN THE @ENTER COMMAND
```

```
072651    %
072651    % ENTRY:      B=BATCH DATAFIELD
072651    %              X=BATCH RT-PROGRAM
072651    %
072651    % EXIT:       MAX CPU TIME IS USED; ABORT JOB
072651    %
072651    % EXIT+1:     MAX CPU TIME NOT USED; THE BATCH JOB WILL CONTINUE
072651    %
072651
072651    CHSMXTIME: X=:D; *AAX BRESL
072653               DO WHILE X:=X.RESLINK><D
072656                  IF X>>="S500S" AND X<<="S500E-5SEMSIZE" THEN
072664                     A:=MXTIME; T:=5670; *RMPY ST DA
072667                     A:=:D-X.L500CPU; *RDCR ADC DD
072672                     A:=:D-X.H500CPU
072674                     IF A<0 THEN EXIT FI
072676                     EXITA
072677                  FI
072677               OD
072700               EXITA
072701
072701    *)FILL
072704
072704
072704
072704    %       5 B P A B O R T
072704    %
072704    % ABORT-BATCH AND ABORT-JOB COMMANDS
072704
072704    5BABORT:  RTREF+"BRESLINK"
072706           DO WHILE A.RESLINK<<"S500S" OR A>>="S500E"
072716              IF A=RTREF THEN EXIT FI
072722           OD; A=:X:=L=:X."LRET":=B=:X.5BRG
072730           A:=1; GO FAR FESC1
072732    *)FILL
072737
072737    % ------------------------------------------------------------------------
072737    %              R L 5 P D E S C   -   1 R L 5 P D E S C
072737    %
072737    % RELEASE THE ND-500 PROCESS DESCRIPTION AFTER THE ND-500 PROCESS HAS BEEN
072737    % TERMINATED. THIS IS CALLED WHEN AN ERROR IS DETECTED BY SINTRAN III MONITOR
072737    % (IN MON.CALL ETC...)
072737    %
072737    % EXECUTED ON MONITOR LEVEL
072737    %
072737    % ENTRY: T=MONITOR LEVEL ROUTINE TO EXECUTE AFTER THE PROCESS IS TERMINATED.
072737    %        X=RT-DESCRIPTION ADDR. (WHEN RL5PDESC)
072737    %
072737    INTEGER POINTER RL5RL
072740    INTEGER MLVPADDR
072741    1RL5PDESC: X:=RTREF
072742    RL5PDESC: A:=L=:"RL5RL"; T=:MLVPADDR
072745           X=:D; CALL FSEMA; GO RL5OUT                    % ANY ND-500 PROCS. RESERVED?
072750           X:=D; CALL MRLCLFIE                            % YES, RESTART ND-100 PROG TO LOGOUT ND-500 PROC
072752           CALL FRWQU; CALL TOEXQU
072754           X.STATUS BONE 11 BZERO 5WAIT=:X.STATUS
072760           X.ACTPRI/\100000+"ALEVB+2"=:X.ACTPRI; *POF
072765           IF X=CURPROG THEN
072770              A/\3773; *TRR PCR
072772              "ERABORT"; *IRW ALEVB DP
```

```
072774              MLVPADDR;  *IRW ALEVB DB
072776           ELSE
072777              "ERABORT"=:X.RTDLGADDR.DPREG
073002              MLVPADDR=:X.DBREG
073004           FI; GO PSTUPR
073005     RL5OUT: X:=D; GO RL5RL
073007     *)FILL
073021
073021
073021     %-------------------------------------------------------------------------
073021     %       E R A B O R T
073021     %
073021     % LOGGOUT PROCESS AFTER ERROR DETECTED BY SINTRAN III MONITOR
073021     % (ERROR IN MONITOR CALL ETC...)
073021     %
073021     % LEVEL 1
073021     %
073021     % ENTRY: B=SINTRAN III MONITOR ROUTINE TO RETURN TO AFTER
073021     %          TERMINATED THE ND-500 PROCESS
073021     %
073021     ERABORT: RTREF=:D; CALL FSEMA; CALL ERRFATAL      % FIND PROCESS DESCRIPTION
073025              A:=B=:X.PDCFLG; ABLOG=:X.SV5FUNC         % USE FUNC=ABLOG TO TERMINATE PROCESS
073031              X.PSTAT BONE 5BRK=:X.PSTAT
073034              X.CDTSEGN BONE 52ESCSET=:X.CDTSEGN
073037              GO FROMESC
073040
073040     %
073040     %       F R O M E S C
073040     %
073040     % ENTRY POINT IN PROGRAMS AFTER ESCAPE
073040
073040     INTEGER POINTER P3STPNT:=STPNT
073041     INTEGER CCODE,CCTYPR
073043     FROMESC:
073043           IF BACKGROUND><0 THEN
073045              RTREF.SEGM/\177400=:T
073051              IF X.ACTSEG/\177400><T THEN A SHZ -10; T+A+1; CALL MMEXY FI  % GET SYSTEM SEG.
073060              5TTIFIELD.FLAGB BZERO 5ESC2SET=:X.FLAGB; "STBEG"=:P3STPNT
073066              IF 5BCHFLAG><1 THEN
073072                 X.TYPRING; *IOF
073074                 IF A NBIT 5BAD AND A BIT M144B OR BIT 5COM THEN T:=-2 ELSE T:=-1 FI
073105                 IF A BIT 5TERM THEN
073107                    X=:CCODE; 0=:X.IN5MSG; X.KSETDV; X:=X.TDRADDR; A=:L; L:=:P
073115                    X:=CCODE.DFOPP; 0=:X.ON5MSG; X.KSETDV; X:=X.TDRADDR; A=:L; L:=:P
073124                    GO FESCX; *)FILL
073137                 FI
073137                 T=:CCODE=:A; X:=:B; CALL SETDV
073143                 A:=CCODE; T:=DFOPP=:B; CALL SETDV; T:=DFOPP; X=:B:=T
073152     FESCX:      *ION
073153              FI
073153           FI
073153     % - CLEAR IN5MSG
073153           RTREF+"BRESLINK"=:X
073156           DO WHILE X:=X.RESLINK><RTREF
073162              IF X>>"5TTST" AND X<<"5TEND" THEN
073170                 A:=X-"DT01R"=:D:=0; T:="5TTSZ"; *RDIV ST
073176                 IF D=0 THEN T:="IN5MSG"; A:=0; CALL XSTDFADDR FI
073203              FI
073203           OD
073204     % - GET PD INTO X
```

```
073204          RTREF=:D; CALL FSEMA; CALL ERRFATAL
073210          IF BACKGROUND><0 THEN
073212             X=:D
073213             IF 5TTIFIELD.FLAGB BIT 5LOGOUT THEN D.PSTAT BONE SOFFLOG=:X.PSTAT FI
073223             X:=D
073224             IF UEFLG BIT 5UECM AND X.PDCFLG=0 THEN  % ESCAPE WHEN IN UECOM?
073231                A BZERO 5UECM =: UEFLG
073233                5SPASSTYPE=:5PASSTYPE
073235                CALL FAR ALLRELEASE
073236                X.CDTSEGN BZERO 52ESCSET =: X.CDTSEGN
073241                X.PSTAT BZERO 5BRK =: X.PSTAT
073244                "STACK"=:5CSTCK
073246                X:=UEXREG                                % RESTORE STATUS BEFORE UECOM
073247                TAD:=UECMRET
073250                D=:P                                     % CONTINUE AFTER UECOM
073251             FI
073251          FI; A:=0; GO FESC1; *)FILL
073277
073277    INTEGER CCSEMAPHORE
073300    FESC1: A=:X.5ERRFLAG; X.CPUDF=:B
073303          CALL FAR ALLRELEASE; MLEV; *MCL PIE
073306          *PIOF
073307          X.CDTSEGN BZERO 52ESCSET=:X.CDTSEGN; X.PSTAT BONE 5BRK=:X.PSTAT
073315          X=:CCSEMAPHORE; X:=X.MESSBUFF; CALL FAR FRQUES       % REMOVE FROM EX.QUEUE OR TIME QUEUE
073320    FESC2: "N500DF"=:B; X:=RTREF; *PION
073324          CALL BRESERVE
073325          IF A<0 THEN
073326             CALL FREXQU; CALL TOWQU; "STUPR"; *IOF; IRW MLEVB DP
073333             MLEV; *MST PIE; MST PID; ION
073337             GO FROMESC; *)FILL
073351          FI; RTREF.ACTSEG/\377=:T; X:=CCSEMAPHORE=:5SEMAPHORE
073357          MLEV; *MST PIE
073361          IF X.5SEG/\377><T THEN
073365             A=:T
073366             IF BACKGROUND><0 THEN RTREF.SEGM/\177400; T+A FI; CALL MMEXY
073375             IF X.RSEGM><0 AND A="S500DF-ZPREG".5RSEGM THEN
073403                CALL SVREINFO
073404                X:="S500DF-ZPREG+5RSEGM"; A:=B+"5RSEGM"; T:=22=:D; *MOVB; JMP *
073413             FI
073413          FI; SG1500=:C5REENT; CALL MMREENT
073416          IF 5SEMAPHORE.PDCFLG><0 THEN
073421             X.SV5FUNC; X:="S500DF-ZPREG"; GO FESC3           % ERROR DETECTED BY SINTRAN III MONITOR
073424          FI; T:="S500DF-ZPREG".5FUNCTION
073426          IF 5STDOM=T THEN N5S3REL ELSE N5REL FI
073434    FESC3: A=:X.5FUNCTION=:5FUNCTION                         % SET NEW FUNCTION CODE
073436          CALL FAR XGETBUFF
073437          GO FAR 5CBREL
073440    *)FILL
073455
073455
073455
073455    %       5 E S C R E S T A R T
073455    %
073455    % MONITOR LEVEL
073455
073455    INTEGER POINTER ESCLL
073456    ESC500:
073456    5ESCRESTART: A:=L=:"ESCLL"; T:="DBPROG"; X:=B; CALL XGTDFADDR
073463          A=:D; CALL FSEMA; GO ESCLL                         % FIND PROCESS DESCRIPTION
073466          *IOF                                               % X=PROCESS DESCRIPTION
```

```
073467          X.CDTSEGN BONE 52ESCSET=:X.CDTSEGN
073472          IF X.PSTAT BIT 5BRK GO MONEN
073475          A BONE 5BRK=:X.PSTAT
073477          "IORESTART"=:MFUNC; *ION
073502          X:=B; CALL S5ESCF
073504          IF X:=RTRES><0 THEN
073506              IF X.TLINK><0 THEN CALL FTIMQU FI
073511              IF X=CURPROG THEN
073514                  "FROMESC"; *IRW ALEVB DP
073516                  X.STATUS BONE 11=:X.STATUS
073521                  "100002+ALEVB"=:X.ACTPRI/\3773; *TRR PCR
073525              ELSE
073526  SYSABORT:       CALL MRLCLFIE; CALL FRWQU; CALL TOEXQU
073531                  X.STATUS BONE 11 BZERO 5WAIT=:X.STATUS
073535                  X.ACTPRI/\100000+"ALEVB+2"=:X.ACTPRI
073541                  *POF
073542                  "FROMESC"=:X.RTDLGADDR.DPREG                    % REG BLOCK IS IN POF
073545                  1=:MTOR                                         % SET RT-D P-REG
073547              FI
073547          FI; GO PMONEN
073550
073550  % ROUTINE TO CHECK IF THE CALLING PROGRAM IS A BACKGROUND PROGRAM
073550  % BEFORE ESCON IS CALLED, IF RT-PROGRAM THEN ESCON IS NOT CALLED
073550  INTEGER XSVA
073551  XONESCAPE: *IOF
073552          A=:XSVA
073553          IF BACKGROUND=0 THEN
073555              A:=XSVA; *ION; EXIT
073560          FI; A:=XSVA; GO ESCON
073562
073562  % ROUTINE TO CHECK IF THE CALLING PROGRAM IS A BACKGROUND PROGRAM
073562  % BEFORE ESCOFF IS CALLED, IF RT-PROGRAM THEN ESCOFF IS NOT CALLED.
073562  XOFFESCAPE: *IOF
073563          A=:XSVA
073564          IF BACKGROUND=0 THEN
073566              A:=XSVA; *ION; EXIT
073571          FI; A:=XSVA; GO ESCOFF
073573
073573  %-----------------------------------------
073573  %       E S C 5 0 N
073573  %
073573  % USER LEVEL - SUBROUTINE TO ESCON IN SINTRAN III
073573  %
073573  % ENTRY; X=TERMINAL DATAFIELD
073573  %        A=FLAGB
073573  %
073573  % EXIT:  EXIT FROM ESCON. A=FLAGB IN TERMINAL DATAFIELD
073573  %
073573  % EXIT +1 : CONTINUE IN ESCON
073573  %
073573  %-----------------------------------------
073573
073573  INTEGER CXREG,CCFLAGB
073575
073575  ESCON:  A=:CCFLAGB:=X.RTRES=:D+"BRESLINK"                        % D: RESERVING PGM; A:THE RESERVATION QUEUE HEA
073601          X=:CXREG                                                % SAVE X
073602          DO WHILE A.RESLINK<<"S500S" OR A>>="S500E"               % SEARCH RESARVATION QUEUE UNTIL N500 PROCESS F
073612              IF A=D THEN X:=CXREG; EXITA FI
073616          OD
073617                                                                  % PROGRAM HAS  RESERVED AN ND500 PROCESS:
```

```
073617          IF A.PSTAT BIT 5BRK THEN X:=CXREG; EXIT FI          % HAS ESCAPE/BREAK BEEN TYPED?
073625  % GO TO 5ESCRESTART ON MONITOR LEVEL WITH ADDR OF DATAFIELD IN X
073625          "5ESCRESTART"; *IRW MLEVB DP
073627          CXREG; *IRW MLEVB DB
073631          MLEV; *MST PID
073633          X:=CXREG; CCFLAGB BONE 5ESCON; EXIT                 % SET ESCAPE ON BEFORE EXIT
073637
073637  *)FILL
073666  *)KILL 7ENDC; 7ENDC=*
073666  *7ENDC/
127153
127153  %
127153  %  H I S T O G R A M  FUNCTIONS
127153  %
127153  IDEFHIST: IF 5HRTP><0 AND A><RTREF THEN EHIUSED; GO FAR ERET FI
127162            RTREF=:5HRTP; CCPUDF=:5HRTB; 0=:5HIFLAG; AD:=5DD3
127170            IF A><0 OR D>>100 OR D=0 GO FAR EEILPAR
127176            A:=D=:5HICHANNELS; AD:=5DD2
127201            IF A><0 OR D=0 GO FAR EEILPAR
127204            A:=D=:5HINTERVAL; AD:=5DD1=:5HISTART
127210  INDEFH:   FOR X:="5HIDATA" TO "5HIOUTSIDE"+1 DO 0=:X.S0 OD
127220            GO FAR OKRET
127221  ESTAHIST: IF 5HRTP><RTREF THEN EHNRESERVED; GO FAR ERET FI
127227            GO FAR ISTA1; *)FILL
127247  *)KILL 7ENDC; 7ENDC=*
127247  *7ENDC/
073666
073666  ISTA1:    *IOF
073667            CALL RHIFEXQ; GO FAR ESTAHIST; *PIOF
073672            GO FAR ISTA2; *)FILL
073676  *)KILL 7ENDC; 7ENDC=*
073676  *7ENDC/
127247
127247  ISTA2:    1=:5HIFLAG
127251            A:="N500DF".5SEMAPHORE-"S500S"=:D:=0; T:=5SEMSIZE; *RDIV ST
127260            X:=HIMESS; T:=5MBBANK; *AAX WANTP; STATX
127264            X:=HIMESS; *STZTX XN5ST
127266            CALL ITO500XQ
127267            *ION
127270            GO FAR OKRET; *)FILL
127300  ESTAHIST: *ION
127301            GO FAR ERET
127302  ISTOHIST: IF 5HRTP><RTREF THEN EHNRESERVED; GO FAR ERET FI
127310            IF 5HIFLAG><0 THEN
127312               *IOF
127313               0=:5HIFLAG; GO FAR ISTO1; *)FILL
127323  *)KILL 7ENDC; 7ENDC=*
127323  *7ENDC/
073676  ISTO1:       CALL RHIFEXQ; *ION
073700               GO FAR OKRET; *)FILL
073703  *)KILL 7ENDC; 7ENDC=*
073703  *7ENDC/
127323            FI; GO FAR OKRET
127324
127324  INTEGER HCOUNT
127325  IREAHIST: IF 5HRTP><RTREF THEN EHNRESERVED; GO FAR ERET FI
127333            0=:ZAREG; MIN ZPREG; 0/\0
127336            A:=5SEMAPHORE-"S500S"=:D:=0; T:=5SEMSIZE; *RDIV ST
127344            A SH 12+5BUSTART=:5BUADR=:X; T:=5BUBANK=:BUADR
127352            -202=:HCOUNT; "5HIDATA"=:L
```

```
127356              FOR HCOUNT DO
127356                 *SWAP SX DL; LDD ,X; AAX 2; SWAP SX DL; STDTX; AAX 2
127364              OD; DBUADR; CALL FAR 5PAATRANS; A=:LOGBADR
127371              X:=5P1; T:=404; GO FAR PERFUNC
127374  IRELHIST:   IF 5HRTP><RTREF THEN EHNRESERVED; GO FAR ERET FI
127402              *IOF
127403              0=:5HIFLAG=:5HRTP; GO FAR IREL1; *)FILL
127423  *)KILL YENDC; YENDC=*
127423  *7ENDC/
073703  IREL1:      CALL RHIFEXQ; *ION
073705              GO FAR OKRET; *)FILL
073710  *)KILL 7ENDC; 7ENDC=*
073710  *YENDC/
127423
127423  *)FILL
127423  @LLIB
127423  @DEV 1
127423  @DEV (S-S-J)ND500-DRIVER-2
```

```
127423    %
127423    %=============================================================================
127423    %       N D 5 0 0 - D R I V E R - 2
127423    %=============================================================================
127423
127423
127423    $LIB CXCPU
127423    %       L O G G I N G   F U N C T I O N S
127423
127423    $SYMBOL LIDLE=2
127423    $SYMBOL LSWPWAIT=4
127423    $SYMBOL LSWPPING=6
127423    $SYMBOL LINMCALL=10
127423    $SYMBOL LACTIVE=12
127423    $SYMBOL LCPU=14
127423
127423    ISTAPRLOG: IF 5HRTP><0 AND A><RTREF THEN ELOGINUSE; GO FAR ERET FI
127432               IF 5HIFLAG><0 AND A><2 THEN EILFUNC; GO FAR ERET FI
127441               AD:=5DD1; IF A><0 OR D>>=CMXACTPROC THEN EILPAR; GO FAR ERET FI
127450               A:=D=:5LOGPROC; 2=:5HIFLAG; RTREF=:5HRTP; CCPUDF=:5HRTB
127460               GO FAR INDEFH
127461
127461    ISTOLOG:   IF 5HRTP><RTREF THEN ELOGNRESERVED; GO FAR ERET FI
127467               0=:5HIFLAG; GO FAR OKRET
127471
127471    IRELLOG:   IF 5HRTP><RTREF THEN ELOGNRESERVED; GO FAR ERET FI
127477               0=:5HIFLAG=:5HRTP; GO FAR OKRET
127502
127502    INTEGER CNW
127503    IPRILOG:   IF 5HRTP><RTREF THEN ELOGNRESERVED; GO FAR ERET FI
127511               IF 5HIFLAG=2 THEN            % PROCESS-LOGG-ONE
127515                 T:=16
127516               ELSE
127517                 0=;"5HIDATA".S2; T:=CMXACTPROC=:X.S3+3 SH 1
127525               FI; T=:L=:CNW; A:="5HIDATA"=:D:=0; T:=5SEMAPHORE.MESSBUFFR
127534               X:=5MBBANK; *MOVPP
127536               T:=CNW; A:=5SEMAPHORE.MESSBUFF=:D:=5MBBANK
127543               CALL FAR 5PAATRANS; A=:LOGBADR
127545               X:=5P2; AD:=5DD1; T SH 1
127550               IF A=0 AND D=0 GO FAR XPFRFUNC
127553    IIPRILOG:  0=:ZAREG; MIN ZPREG; 0/\0
127556               GO FAR PFRFUNC
127557
127557    ISTLAPR:   IF 5HRTP><0 AND A><RTREF THEN ELOGINUSE; GO FAR ERET FI
127566               IF 5HIFLAG><0 AND A><3 THEN EILFUNC; GO FAR ERET FI
127575               RTREF=:5HRTP; CCPUDF=:5HRTB; 3=:5HIFLAG; GO FAR INDEFH
127604    *)FILL
127625
127625    ISTAMLOG: IF 5MLOG><RTREF AND A><0 THEN ELOGINUSE; GO FAR ERET FI
127634              AD:=5DD1
127635              IF A >< 0 OR D >< 0 THEN
127640                 CALL FAR PCHSYSTEM; A:= -1
127642              ELSE
127643                 5SEMAPHORE-"S500S"; A=:D:=0; T:=5SEMSIZE; *RDIV ST
127651              FI
127651              A=:5MLOPROC; RTREF=:5MLOG
127654              T:=5BUBANK; X:=5BUSTART=:D; A:=2000; D+A
127661              DO WHILE X><D
127663                 *STZTX
127664                 X+1
```

```
127665              OD; GO FAR OKRET
127667    IPRIMLOG: IF 5MLOG><RTREF THEN ELOGNRESERVED; GO FAR ERET FI
127675              DBUSTART; CALL FAR 5PAATRANS; A=:LOGBADR
127700              X:=5P1; T:=3000; GO FAR IIPRILOG
127703    ISTOMLOG: IF 5MLOG><RTREF THEN ELOGNRESERVED; GO FAR ERET FI
127711              0=:5MLOG=:5MLOPROC; GO FAR OKRET
127714    *)FILL
127731
127731    ILI5TQU: K:=1; *IOF
127733           AD:=5ATIME; T:=BUADR; X:=5BUADR; *STDTX
127737           X+2=:D; X:=TIM2LINK; GO ILI5F
127743
127743    ILI5EXQ: K:="0"; *IOF
127745           CCPUDF.C5PROC; T:=BUADR; X:=5BUADR; *STATX
127752           X+1=:D; X:=CCPUDF.MAILINK
127756    ILI5F: T:=5MBBANK
127757           DO WHILE -1><X
127762              X=:L; *LDATX X5SND
127764              IF A><-1 THEN
127767                 X:=D; T:=BUADR; *STATX                          % PROCESS NUMBER
127772                 T:=5MBBANK; L=:X:=D
127775                 IF K THEN                                       % LIST-TIME-QUEUE
127777                    *AAX D5TIM; LDDTX                            % START-TIME
130001                    X:=:L; T:=BUADR; *STDTX 10
130004                    X+1
130005                 ELSE                                            % LIST-EXEC-QUEUE
130006                    X=:L
130007                    *AAX XADPR; LDXTX
130011                    X.5PRIORITY; X:=D; T:=BUADR; *STATX 10       % PRIORITY
130015                 FI; X+2=:D
130017              FI; X:=L; T:=5MBBANK; *LDXTX XLNK2
130022           OD; -1; X:=D; T:=BUADR; *STATX; ION
130030           A:=X/\1777+1 SH 1=:T; X:=5P1; GO FAR IIPRILOG
130037    *)FILL
130042
130042
130042    %       R H I F E X Q
130042    %
130042    % REMOVE HISTOGRAM MESSAGE FROM THE N500 EXECUTION QUEUE
130042    %
130042    INTEGER 5HILREG=?
130042    RHIFEXQ: CCPUDF=:B
130044               X:=HIMESS; GO FAR FRQUES
130046    *)FILL
130050    %
130050    %       5 0 0 H I S T
130050    %
130050    % SUBROUTINE CALLED EACH BASIC TIME UNIT
130050    %
130050    % MONITOR LEVEL; CALLED FROM ICLCK
130050    %
130050    INTEGER 5HSTOP,5HCMESS,5HBRG=?,5HILREG=?
130052    500HIST: AD:=ATIME=:"N500DF".5ATIME; A:=L=:5HILREG
130057           0=:5HSTOP; X:=:B; X=:5HBRG
130062           DO WHILE X:=TIM2LINK><-1                            % SEARCH TIME QUEUE AND START PROCS.
130066              X=:5HCMESS
130067              T:=5MBBANK; *AAX D5TIM; LDDTX; AAX -D5TIM
130073              A=:L:=5ATM2-D:=5ATM1; *RADD ADC CM1 SL DA
130100              IF A<0 GO 500H1                                  % NO MORE PROCS TO START
130101              T:=5MBBANK; *LDDTX XLNK1
```

```
130103              AD=:TIMLINK                                    % REMOVE FROM TIME QUEUE
130104              *AAX XADPR; LDXTX
130106              IF X.RTRES><0 AND X.MAGNO><0 THEN              % PROCESS IS STILL ALIVE
130112                 X.CPUDF=:B
130114                 IF 5HSTOP=0 THEN CALL TER500; 0/\0; MIN 5HSTOP FI
130121                 X:=5HCMESS; CALL FR5TMQU                    % REMOVE FROM TIME QUEUE
130123                 A:=0; CALL SPITMQ; CALL IT0500XQ            % START PROC
130126                 "N500DF"=:B
130130              ELSE
130131                 X:=5HCMESS; CALL FAR FRQUES                          % REMOVE FROM TIME QUEUE
130133              FI
130133           OD
130134     500H1: IF 5HSTOP><0 THEN CALL LOWACT500 FI                        % RESTART ND-500 IF STOPPED
130137           CALL FAR PIONPIOF; GO 500H2; *)FILL
130153
130153     INTEGER 5HILREG
130154     INTEGER 5HBRG
130155
130155     500H2: A:="S5CPUDF"=:B
130157           DO WHILE B <<= "E5CPUDF"
130162              IF CPUAVAILABLE><0 THEN
130164                 T:=HDEV+RSTA; *IOXT
130167                 IF A BIT 5ILOCK AND A NBIT 5POWOF THEN
130173                    A:=B+2000=:X; X.C5PROC                    % CAHCE MISS!!!!!
130177                    IF C5PROC >< -1 THEN
130203                       A*5SEMSIZE+"S500S"=:X; AD:=X.500TU; D+1; A:=A+C; AD=:X.500TU
130212                    FI
130212                 FI
130212                 IF 5HIFLAG=3 THEN                                    % PROCESS-LOGG-ALL
130216                    IF B = "S5CPUDF" THEN
130221                       MIN "5HIDATA".S5; *SKP; MIN 4,X; JMP *1
130226                       X:=SWMSG; T:=5MBBANK; *LDATX XN5ST        % A:=SWMSG.N5STATUS
130231     %                 %*AAX SWPFU; LDXTX                        % X:=X.SWPFUNC
130231     %                 %IF A><PSWWAIT OR X><NEWSWAP THEN
130231                       IF A><PSWWAIT THEN                        % SWAPPER FREE?
130234                          MIN "5HIDATA".S1; *SKP; MIN ,X; JMP *1
130241                       FI
130241                    FI
130241                    IF C5PROC><-1 THEN
130245                       A SH 1+"5HIDATA"+6=:X
130251                       *MIN 1,X; SKP; MIN ,X; JMP * 1
130255                    FI
130255                 FI
130255              FI; A:=5CPUDFSZ; B+A
130257           OD
130260           IF 5HRTP=0 OR 5HIFLAG=0 GO FAR 5HIRET
130265           5HRTB=:B; T:=HDEV+RSTA; *IOXT
130272           IF A NBIT 5ILOCK OR A BIT 5POWOF GO FAR 5HIRET
130276           IF 5HIFLAG=1 THEN                                  % HISTOGRAM
130302              2000=:D
130304              X:=HIMESS; T:=5MBBANK
130306              X+D; *LDATX XN5ST; RSUB SD DX; LDATX XN5ST        % CACHE MISS BEFORE READING STATUS
130312              IF A<<=ANSWER THEN
130315                 T:=5MBBANK; *AAX WANTP; LDXTX                % X:=HIMESS.WANTPRO
130320                 X:=ADRPROC(X)
130321                 IF X.5RUNSTATUS = ACTIVE THEN
130325                    A:=MSGN500; X:=HIMESS; T:=5MBBANK; *STATX XN5ST
130331                    CALL LOWACT500; CALL FAR PIONPIOF
130333                 FI; GO 5HIRET; *)FILL
130353                 INTEGER POINTER P5HBRG:=5HBRG
```

```
130354          FI; GO 5HIRET
130355        FI
130355        IF A=2 THEN                                   % PROCESS-LOGG-ONE
130360          MIN "5HIDATA".S1; *SKP; MIN ,X; JMP * 1
130365          IF C5PROC=5LOGPROC THEN
130371            LACTIVE
130372          ELSE
130373            X:=ADRPROC(5LOGPROC)
130375            IF X.RTRES=0 OR X.5RUNSTATUS><ACTIVE AND A><INMCALL THEN
130406              LIDLE
130407            ELSE
130410              2000=:D
130412              X:=X.MESSBUFF; T:=5MBBANK
130414              X+D; *LDATX XN5ST; RSUB SD DX; LDDTX XN5ST % A=STATUS; D=SENDER; CAHCE MISS BEFORE READING
130420              IF A=SWPWAIT THEN
130423                LSWPWAIT
130424              ELSE
130425                IF A=SWPPING THEN
130430                  LSWPPING
130431                ELSE
130432                  X:=ADRPROC(D)                       % D:=SENDER
130434                  IF X.5RUNSTATUS=ACTIVE THEN
130440                    LCPU
130441                  ELSE
130442                    IF A=INMCALL THEN
130445                      LINMCALL
130446                    ELSE LIDLE
130450          FI FI FI FI FI FI
130450          X:="5HIDATA"+A
130452          *MIN 1,X; SKP; MIN ,X; JMP * 1
130456          GO 5HIRET; *)FILL
130464          INTEGER POINTER P5HILREG:=5HILREG
130465        FI
130465  5HIRET: P5HBRG=:B; P5HILREG=:P                      % EXIT
130471  *)FILL
130471  *)KILL YENDC; YENDC=*
130471  */ENDC/
073710
073710  PIONPIOF: *PION; PIOF; EXIT
073713  RHIFRET: *PION; EXIT
073715  PFRFUNC: MLEV; *PON; MST PIE
073720        GO FAR RFUNC
073721
073721  % ROUTINE TO MOVE DATA FROM THE "MON 60" COMMUNICATION BUFFERS OR
073721  % FROM THE ND-500 MESSAGE TO THE USER AREA
073721  %
073721  % ENTRY:      T=NUMBER OF BYTES TO MOVE
073721  %             X=ADDRESS IN USER'S AREA
073721  %
073721  XPFRFUNC: MLEV; *PON; MST PIE
073724            T=:D BONE 16; LOGBADR; CALL 5ALTON; *MOVB; JMP *; BSET ZRO
073733            MLEV; *MCL PIE; POF
073736            GO FAR INDEFH
073737
073737  %-------------------------------------------------------------------------------
073737  %             S 5 T S L T Y P E
073737  %
073737  % SET TIMESLICE TYPE FOR ND-500 PROCESS
073737  %
073737  % ENTRY:      A=TIMESLICE TYPE
```

```
073737    %              X=PROCESS NUMBER
073737    %
073737    $$TSLTYPE:
073737            *PIOF
073740            X=:D:=ADRPROC(X); A=:T
073743            IF X.PSTAT BIT SLICE THEN
073746               A:=5TSLSTATUS(D)/\1777; T SH 12
073752               A\/T BONE 5BRKF=:5TSLSTATUS(X)
073755            FI; *PION
073756            EXIT
073757
073757
073757    % ===========================================================================
073757    %              5 S 1 T S L T Y P E   -   5 R 1 T S L T Y P E
073757    %
073757    % SUBROUTIEN TO SET AND RESET TIMESLICE TYPE
073757    %
073757    % ENTRY:       X=RT-DESCRIPTION ADDRESS
073757    %              A=NEW SLICE TYPE (0-7)
073757    %
073757    % REGISTER USED:              T,A,D,L
073757    %
073757
073757    INTEGER POINTER L2RG
073760    INTEGER CTYP,PROG,3SLICE
073763
073763    $$1TSLTYPE: K:="0"; GO FELLX
073765    $R1TSLTYPE; K:=1
073766    FELLX: *IOF
073767            A=:CTYP:=L=:"L2RG"; X=:PROG; D=:3SLICE
073774            CALL GTSLPINDEX; GO UT
073776            IF T:=TSLSTATUS(X) NBIT 5NOSLICE THEN
074001               1=:3SLICE
074003               IF K THEN
074005                  A:=T SHZ -3; D:=T          % SET CURRENT = SAVED TYPE
074010               ELSE
074011                  A:=CTYP SH 12; D:=T SH 3  % SAVE CURRENT, SET NEW CURRENT TYPE
074015               FI; A/\16000:=:T/\1777:=:D/\160000\/T\/D BONE 5ESCF=:TSLSTATUS(X)
074026            FI
074026    UT:     X:="L2RG"=:L:=PROG; A:=3SLICE; *ION
074033            EXIT
074034
074034
074034
074034    % ***************************************************************************
074034    %       N 5 0 0 - S C H E D U L E R
074034    % ***************************************************************************
074034
074034    %       RT-PROGRAM TO SCHEDULE N500 PROGRAMS
074034    %       (SUBROUTINE TO THE TIMER RT-PROGRAM)
074034    %       INTERVALL:              0.6 SEC.
074034
074034    %       THE PROCESS-DESCRIPTION'S TIME-USED IS UPDATED BY THE
074034    %       MONITOR LEVEL AT EACH BASIC TIME UNIT
074034
074034    INTEGER POINTER TSLREG
074035    N500SCHEDULER: A:=L=:"TSLREG"; MLEV; *MCL PIE; POF
074042            GO FAR SCHE0; *)FILL
074056    *)KILL 7ENDC; 7ENDC=*
074056    *7ENDC*
```

```
130471
130471    INTEGER SCEXX
130472    SCHED:
130472    *"8N500 5MUCP
"130472
130472            IF "S5CPUDF".CPUAVAILABLE><0 THEN
130475               T:=X.HDEV+RSTA; *IOXT
130500               IF A BIT 5ILOC AND X.C5STAT NBIT BHPFAIL GO NN5S1
130505            FI; GO FAR SRET
130506
130506    *"8N500
"130506
130506    *)FILL
130510
130510    INTEGER CTSLCLASS=?
130510    INTEGER CTSLPROC=?
130510    INTEGER CTSLSTATUS=?
130510    INTEGER CINDEX=?
130510
130510    NN5S1:
130510    *"8N500 -5MUCP
"130510             D=:TS5STOP
130511    *"8N500 5MUCP
"130511             I=:CINDEX; X:="S500S+5SEMSIZE"
130514    %%       DO WHILE X<<"S500E"
130514    DOX:     IF X>>="S500E" GO FAR EDOX
130517    %%          IF X.RTRES><0 AND X.PSTAT BIT SLICE AND X.5PRIORITY><5COMPRIOR THEN
130517              IF X.RTRES=0 OR X.PSTAT NBIT SLICE OR X.5PRIORITY=5COMPRIOR GO FAR EFIX
130530                 X=:B=:CTSLPROG
130532                 IF LPRCOUNT<0 THEN
130534                    MIN LPRCOUNT; GO CONWAIT
130536                    T:=5MBBANK; X:=MESSBUFF; SUSPMESSTAT; *STATX XN5ST
130542                 FI; 5TSLSTATUS(CINDEX)=:CTSLSTATUS SHZ -7CUTY/\7=:CTSLCLASS
130550                 IF CTSLSTATUS BIT 5BRKF THEN                % WAITING FOR BREAK PRIORITY?
130553                    A BZERO 5BRKF=:CTSLSTATUS                % CLEAR BREAK FLAG
130555                    IF 5PRIORITY<=TSLLOWLG THEN              % CAN PRIORITY BE INCREASED
130561                       TSLBRKELEM(CTSLCLASS)                 % GET BREAK ELEMENT
130563    INBRKTEST:        A=:D:=CTSLSTATUS/\177740\/D=:CTSLSTATUS
130570                       L500CPU=:5TSLNTIME(CINDEX)
130573                       GO SETALL; *)FILL
130610
130610            INTEGER CTSLCLASS
130611            INTEGER CTSLPROC
130612            INTEGER CTSLSTATUS
130613            INTEGER CINDEX
130614
130614                       FI
130614                 FI
130614                 A:=L500CPU-5TSLNTIME(CINDEX)=:D:=0
130621                 T:=TSLTUNIT; *RDIV ST
130623                 IF A+5TSLCOUNTA(X)<0 GO CONWAIT             % COMPUTE CPU TIME USED
130625                 L500CPU=:5TSLNTIME(X)                       % TIMESLICE FINISHED?
130627    FINNEW:      CTSLSTATUS/\37=:X                           % YES, SET NEW CPU TIME
130632                 A:=CTSLSTATUS/\177740\/TSLNEXTAB(X)         % TIMESLICE ELEMENT
130635                 A=:CTSLSTATUS/\37=:D                        % FIND NEXT ELEMENT IN CHAIN
130640
130640    SETALL:      IF TSLTIMTAB(D)=:T-TSLHTIME>=0 THEN         % HASH ELEMENT?
130645                    L500CPU/\TSLHASHM+T                      % YES; HASH WITH TIME USED
130650                 ELSE
130651                    A:=T
```

```
130652                  FI; A-=:5TSLCOUNTA(CINDEX)                    % SET TIMSLICE COUNTER
130655                  *IOF
130656                  T:=TSLPRITAB(D)=:5PRIORITY
130661                  IF TSLLPRITAB(CTSLCLASS)>=T THEN
130665                    A=:HTSLLOWPRI; PSTAT BONE 5LTSLPRI
130670                  ELSE
130671                     PSTAT BZERO 5LTSLPRI
130673                  FI; A=:PSTAT
130674                  X:=MESSBUFF; CPUDF=:B; CALL CHEXQ; GO TSL5NEXT
130701  *IBN500 -5MUCP
 130701                  IF TS5STOP=0 THEN
130703                    MIN TS5STOP; CALL TER500; GO ESCED; *TRR CCLR
130707                  FI
130707  *IBN500 5MUCP
 130707                  X:=CTSLPROC.MESSBUFF; CALL IFM500XQ; CALL XITO500XQ
130713  TSL5NEXT:       *ION
130714                  CTSLSTATUS=:5TSLSTATUS(CINDEX)
130717  UNWAIT:         X:=CTSLPROC
130720  %%          FI; MIN CINDEX; X+5SEMSIZE
130720  EPIX:       MIN CINDEX; X+5SEMSIZE
130722  %%      OD
130722          GO FAR DOX
130723
130723  EOOX:
130723  *IBN500 -5MUCP
 130723           IF TS5STOP><0 THEN
130725             *IOF
130726             "S5CPUDF"=:B; GO FAR SCE05
130731          FI
130731  *SCE06=*
130731          GO FAR SRET; *)FILL
130757  * IBN500 5MUCP
 130757
130757  ESCED: CALL RSTARTALL; GO FAR SRET; *)FILL
130763  *)KILL YENDC; YENDC=*
130763  */ENDC/
074056
074056  SRET:   MLEV; *MST PIE; PION
074061          GO TSLREG
074062  SCE05:  CALL LOWACT500; CALL FAR PIONPIOF
074064          *JMP I (SCE06
074065  *)FILL
074070
074070
074070
074070  % ************************************************************************
074070  %       N 5 0 0 - A B S T R A N S   P R O G R A M
074070  % ************************************************************************
074070
074070  %       RT-PROGRAM TO READ/WRITE PAGE FROM/TO DISK TO/FROM MEMORY
074070
074070  INTEGER ARRAY ABSLI:=(XABSFU,XSWMEMA,XABLOC,XABLNO)
074074  INTEGER XSDUNIT(2)              % LOGICAL UNIT OF SWAP DEVICE
074076  INTEGER XABSFU                  % FUNCTION IN ABSTR
074077  DOUBLE XSWMEMA                  % MEMORY ADDRESS OF DISC TRANSFER
074101  INTEGER XABLOC(2)               % MASS STORAGE (DISC) ADDRESS
074103  INTEGER XHABLC                  % DUMMY (DO NOT REMOVE!)
074104  INTEGER XABLNO                  % NUMBER OF SECTORS TO TRANSFER
074105  INTEGER SWHSTAT                 % SWAP DEVICE HARDWARE STATUS
074106  INTEGER CKFLIP                  % ERROR INDICATOR
```

```
074107    DOUBLE DSWMSG                    % ADDRESS TO SWAP "ABSTR" PARAMETERS IN SWMSG
074111
074111    5SWRT: *PIOF
074112           ADRPROC(0)=:B; X:=RTREF
074116           CALL BRESERVE; IF A<0 THEN CALL ERRFATAL FI
074121           X:="5CPU1":=:B
074123           A:=SWMSG+"SWPINFO"=:D:=5MBBANK; AD=:DSWMSG
074130           *PION
074131    % - B-REG WILL BE SET BY ACTSWAPPER WHEN MULTIPLE ND-500
074131           DO
074131    INRT:      CALL FAR SETIOWAIT
074132             AD:=DSWMSG; T:=11=:L:="XSDUNIT"; *MOVPN
074137             T:=XSDUNIT; "ABSLI"; *MON 131
074142             IF A>=0 THEN T:=0=:A=:D ELSE A=:SWHSTAT:=SWDERR=:D:=0; T:=1 FI
074154             T=:CKFLIP; X:=SWMSG; *PIOF
074157             CALL MONICO; CALL TER500; GO ERRINRT
074162             X:=ADRPROC(0); 0=:X.F5BUFF; LTTMR=:TMR
074167             CALL LOWACT500; *PION
074171           OD
074172    ERRINRT: CALL RSTARTALL; X:=ADRPROC(0); *PION
074176             0=:X.F5BUFF; GO INRT
074200
074200    *)FILL
074220
074220
074220
074220    % ***************************************************************************
074220    %       N 5 0 0 T M R
074220    % ***************************************************************************
074220
074220    %       TIME-OUT ROUTINE FOR THE N500
074220    %       CALLED IN PIOF
074220
074220
074220    INTEGER POINTER TMRET
074221    N500TMR: *POF
074222           A:=L=:"TMRET"; GO FAR 5TMR1; *)FILL
074226    *)KILL 7ENDC; 7ENDC=*
074226    *7ENDC/
130763
130763    INTEGER CCERR,CTER5
130765    5TMR1: IF CPUAVAILABLE=0 GO FAR TMEXIT
130770           IF C5STAT/\3=3 GO 5TMRA
130775           177377; CALL CLESTATUS
130777           IF A BIT 5POWOF THEN
131001    5TMRA:     TTMR=:TMR; GO FAR TMEXIT
131004           FI
131004           IF T:=C5STAT BIT BHPFAIL THEN
131007              IF T NBIT BCSLPFAIL THEN A BONE POWDOWN; CALL RSTARTALL FI
131013              177177; CALL CLESTATUS; GO FAR TMEXIT
131016           FI; IF A/\ 720 ><0 GO N5ABORT
131020           IF TMRXQ=WATCHDOG THEN
131024              0=:TMR
131025    N5ABORT:   N5TIMOUT
131026    5TMR2:     A=:CCERR; 0=:TMRXQ
131030              CALL RSTARTALL; GO FAR TMEXIT
131032           FI
131032           IF MAILINK=-1 OR FERROR><0 GO FAR TMEXIT
131041           0=:CTER5
131042           X:=SWMSG; T:=5MBBANK; *LDDTX XLNK2                    % D:=SWMSG.N5STATUS
```

```
131045            *AAX SWPIN; LDATX
131047            IF A><0 AND D=PSW1WAIT THEN                          % WAITING FOR ND-100?
131053               T:=5MBBANK; X:=A; *AAX XADPR; LDXTX
131057               IF X.5RUNSTATUS=5IDLE OR X.RTRES=0 THEN           % ND-100 PROC NOT OK?
131065                  X:=SWMSG=:CTER5; CALL TER5000; GO 5TMR2
131071                  A:=0; CALL EMONICO
131073               FI
131073            FI
131073            IF CTER5=0 THEN CALL TER500; GO 5TMR2; FI
131077            X:=WATCHDOG=:TMRXQ; T:=5MBBANK; MSGN500; *STATX XN5ST
131104            CALL IT0500XQ; GO FAR TMX2; *)FILL
131124    *)KILL  YENDC; YENDC=*
131124    */ENDC/
074226
074226    TMX2:   LTTMR=:TMR; CALL LOWACT500; *PION
074232    TMEXIT: GO TMRET
074233
074233
074233    %       M L O W A C T
074233    %
074233    % ACTIVATE N500 FROM THE MONITOR LEVEL
074233    % CALLED IN "PIOF"
074233    % RETURNS TO MONEN
074233
074233    MLOWACT: CALL LOWACT500; *PION
074235               GO MONEN
074236
074236    %       L O W A C T 5 0 0
074236    %
074236    % ROUTINE TO ACTIVATE N500 FROM LOWER LEVELS THEN THE DRIVER LEVEL
074236    %
074236    % THE ROUTINE MUST BE CALLED IN "PIOF"
074236    %
074236    LOWACT500:
074236            A:=B; *IRW LV12B DB
074240            "5STDRIV"; *IRW LV12B DP
074242            LV12; *MST PID; EXIT
074245    5STDRIV: *POF
074246            GO FAR ENTDRIV
074247    *)FILL
074255    *)KILL 7ENDC; 7ENDC=*
074255    */ENDC/
131124
131124
```

```
131124
131124    %
131124    % SUBROTINE TO THE SINTRAN III POWER FAIL ROUTINE ON LEVEL 14
131124    %
131124    INTEGER 5PFFL
131125    5PF1:   1=:5PFFL; GO 5PF
131130    5PF2:   2=:5PFFL; GO 5PF
131133    5PF3:   3=:5PFFL
131135    5PF:    "S5CPUDF"=:B
131137            DO WHILE B <<= "E5CPUDF"
131142               IF CPUAVAILABLE><0 THEN
131144                  IF 5PFFL = 1 THEN
131150                     0=:C5PWF
131151                  ELSE IF A = 3 THEN
131155                     C5STAT BONE BHPFAIL=:C5STAT
131160                  FI; FI
131160                  IF C5PWF=0 THEN
131162                     T:=HDEV+RSTA; *IOXT
131165                     IF A BIT 5ILOC THEN
131167                        T+"TERMI-RSTA"; *IOXT
131171                        FOR X:=-10 DO; OD
131173                        T+"RSTA-TERMI"; *IOXT
131175                        IF A BIT 5ILOC THEN
131177                              GO NCPUPF      % NEXT CPU IF THIS ONE DID NOT STOP
131200                        FI
131200                     FI; 1=:C5PWF
131202                     T:=HDEV+LCON; A:=10; *IOXT
131206                     T+"RSTA-LCON"; *IOXT
131210                     A BONE 5POWOF; T+"LSTA-RSTA"; *IOXT
131213                     A:="0"; T+"LCON-LSTA"; *IOXT
131216                     A:=400; *IOXT
131220                     T+"SLOC-LCON"; *IOXT
131222                     T+"TERMI-SLOC"; *IOXT
131224                  FI
131224               FI
131224    NCPUPF.    A:=B+5CPUDFSZ=:B
131227            OD; EXIT
131231    *)FILL
131233
131233    % ***************************************************************************
131233    %       N 5 0 0
131233    % ***************************************************************************
131233
131233    %       NORD-500 COMMUNICATION DRIVER KERNEL
131233    %       INTERRUPT LEVEL 12
131233    %
131233
131233
131233
131233    INTEGER NSCANN
131234    ENTDRIV: 1=:NSCANN; GO IND01
131237    N500:
131237    DO1:    DO
131237               0=:NSCANN
131240    IND01:     IF C5STAT BIT BHPFAIL GO CALLID12
131243               177377; CALL CLESTATUS
131245               IF A/\720><0 THEN
131247                  *TRR CCLR
131250                  IF A BIT 5PFAIL THEN
131252                     C5STAT BONE BHPFAIL BZERO BCSLPFAIL=:C5STAT
```

```
131256                      TTMR=:TMR; KPOWDOWN
131261                   ELSE IF A BIT 5DMAERR THEN N5DMAERR
131265                   ELSE N5IERR
131267                   FI; FI
131267   INN500:         CALL HRSTARTALL; GO CALLID12
131271               FI
131271   TOPQ:       *TRR CCLR
131272               X:=MAILINK
131273               DO WHILE X><-1
131276                  X=:MESSLINK
131277   INDO:          T:=5MBBANK; *LDATX XN5ST
131301                  IF A=ANSWER THEN
131304                     IF X=HIMESS THEN CALL HISTSAMPLE
131310                     ELSE IF X=WATCHDOG THEN
131314                              0=:TMRXQ; CALL IFM500XQ; LTTMR=:TMR
131320   *)N500 5MUCP
131320                      ELSE CALL DECOMESS
131322                     FI; FI
131322                  ELSE IF A=ERRANSWER THEN CALL DECOERRMESS
131327                  ELSE IF A=WAITING AND NSCANN=0 THEN
131335                            CALL TER500; GO 5EN500; MIN NSCANN; *TRR CCLR
131341                            GO INDO
131342                  ELSE IF A>>100 THEN CALL RRTWT
131347                  FI; FI; FI; FI
131347   NXTMSG:        X:=MESSLINK; T:=5MBBANK; *LDXTX XLNK2
131352   EOD:        OD; CALL IACT500
131354   CALLID12: CALL ID12
131355             OD
131356   5EN500: ESPTIMOUT; GO INN500
131360   *)FILL
131402
131402
131402   %       D E C O M E S S
131402   %
131402   % DECODE "ANSWER-MESSAGE" FROM N500
131402   % THE ACTUAL MESSAGE IS FOUND IN MESSLINK
131402   %
131402   % DRIVER LEVEL
131402   %
131402   % X=ACTUAL MESSAGE (=MESSLINK)
131402   % B=N500 DATAFIELD
131402   %
131402
131402   INTEGER POINTER RETO
131403   DECOMESS: A:=L=:"RETO"
131405          T:=5MBBANK; *AAX SPFLA; LDATX; AAX -SPFLA
131411          IF A><0 THEN A=:P FI
131413          *LDATX XMICF
131414          IF A=3MONCO OR A=3TRACO OR A=3START OR A=3WMONCO THEN
131430             T:=5MBBANK; *AAX STOPR; LDATX; AAX -STOPR
131434             IF A=MOCALL THEN CALL MCHANDLE
131440             ELSE IF A=TRAPCODE THEN CALL TRAPDECODER
131445             ELSE FATERR; T:=5MBBANK; *STATX XN5ST
131451                  CALL RRTWT
131452             FI; FI
131452          ELSE CALL RRTWT
131454          FI; GO RETO
131455
131455   *)FILL
131461
```

```
131461    %       D E C O E R R M E S S
131461    %
131461    % DECODE "ERROR-ANSWER" MESSAGES FROM N500
131461    % THE ACTUAL MESSAGE IS FOUND IN MESSLINK
131461    %
131461    % DRIVER LEVEL
131461    %
131461    % ENTRY:       X=ACTUAL MESSAGE
131461    %              B=N500 DATAFIELD
131461    %
131461    INTEGER POINTER RET1
131462    DECOERRMESS: A:=L=:"RET1"
131464           T:=5MBBANK; *AAX TRAPN-1; LDDTX                % D=TRAPNO
131467           *AAX 1-TRAPN; LDATX XMICF                      % A=MICFUNC
131471    @LR;
131471           IF D=46 AND A=3MONCO OR =3RMED OR =3WMEP OR =3RMEP OR =3WMONCO
131511              OR =3PHSREAD OR =3PHSWRITE OR A=3WMED GO ITRAPDECODER
131524    @LR;
131524           CALL RRTWT; GO RET1
131526
131526    %       T R A P D E C O D E R
131526    % DECODE "TRAP-MESSAGES" FROM N500
131526    %
131526    % DRIVER LEVEL
131526    %
131526    % ENTRY:       X="TRAPPED" MESSAGE
131526    %              B=N500 DATAFIELD
131526    %
131526
131526    TRAPDECODER:
131526           A:=L=:"RET1"; T:=5MBBANK
131531           *AAX TRAPN-1; LDDTX; AAX 1-TRAPN               % D=TRAPNO
131534           IF D>53 THEN
131537              ILTRAP; T:=5MBBANK; *STATX XN5ST
131542              CALL RRTWT                                  % UNKNOWN TRAP
131543           ELSE IF D = 46 THEN                            % PAGE FAULT
131547    ITRAPDECODER: IF X>< SWMSG THEN
131552                     IF INITFLAG BIT BRESPLACE GO ITRPERR
131555                     MSWPFAULT SHZ 10+D; T:=5MBBANK; *AAX TRAPN; STATX; AAX -TRAPN
131564                     CALL 5ACTSWAPPER
131565                  ELSE
131566                     EPFINSWAP; CALL HRSTARTALL
131570                  FI
131570           ELSE
131571    ITRPERR:    CALL RRTWT
131572           FI;FI; GO RET1
131573    *)FILL
131602
131602    %       S W P D E C O D E R
131602
131602    %       MESSAGE DEMANDING SWAPP IS MARKED "SWAPWAIT" IN STATUS
131602    %       THE SWAPPER IS ACTIVATED WITH PAGE-FAULT INFO IN THE MESSAGE
131602    %       AND THE SWAPPED PROCESS IS MARKED "SWAPPING"
131602
131602
131602    SWPDECODER:
131602           T:=5MBBANK; *AAX SWPFU; LDATX
131605           IF A >> SWFMAX GO FAR ESWPFATAL
131610           A GOSW FAR ESWPFATAL, LNEWSWAP, FAR LSWPAGE, FAR LPRSUSPEND, FAR LALLOPAGE, FAR LDATREADY, FAR LCLT
131620
```

```
131620    INTEGER SAVB,CSWPM
131622    CNEWSWAP:
131624            T:=5MBBANK; X:=SWMSG; *AAX SWPIN; LDXTX                    % X:=SWMSG.SWPINFO
131626            X=:CSWPM
131627            IF X><0 THEN
131630                CALL ONESCAPE; GO SWPDT
131632                T:=5MBBANK; X:=SWMSG; *AAX SWPST; LDATX                % A:=SWMSG.SWPSTAT
131636                A=:D
131637                IF A><0 THEN
131640                    *AAX SWPIN-SWPST; STZTX
131642                    X:=CSWPM; *AAX SPFLA; LDATX                        % D=:X.SWPINFO
131645                    IF A><0 THEN
131646                        *STZTX                                         % A:=SWMSG.SWPINFO.SPFLAG
131647                        D=:A                                           % D=:X.SPFLAG
131650                        X:=CSWPM; CALL EMONICO                         % SWPSTAT, ERROR CODE
131652                    ELSE
131653                        D=:A
131654    SWPDT:              X:=CSWPM; *STATX XN5ST                         % SWPSTAT, ERROR CODE
131656                        CALL RRTWT
131657                    FI
131661                ELSE
131660                    X:=CSWPM; *LDATX XN5ST
131662                    IF A=SWPPING THEN
131665                        T:=5MBBANK; *LDDTX X5SIZ                       % A:=SWMSG.SWPINFO.N5STATUS
131667                        IF D=3SWMESS THEN                              % D:=SWMSG.SWPINFO.MICFUNC
131672                            ANSWER; T:=5MBBANK; *STATX XN5ST           % ANSWER=:X.N5STATUS
131676                            CALL RRTWT
131677                        ELSE
131700                            MSGN500; T:=5MBBANK; *STATX XN5ST          % MSGN500=:X.N5STATUS
131705                            IF 3START=D THEN
131707                                3TRACO; *STATX XMICF        % 3TRACO=:X.MICFUNC
131710                            ELSE
131711                                IF 3WMONCO=D THEN          % WRITEBACK BUFFER RESTART?
131713    % RESTART A RESTART-AFTER-MONITOR-CALL-WITH-BUFFER-WRITEBACK MESSAGE, MICFUNC26.
131713    % BUFFER ADDRESS AND LENGHT MUST BE RESTORED SINCE THESE HAVE BEEN OVERWRITTEN
131713    % BY IRAPINFO. TO IDENTIFY THE MONITOR CALL WE USE THE WRITEBACK MASK SINCE THE
131714    % MONITOR CALL NUMBER HAS BEEN OVERWRITTEN BY THE FIRST PART OF FUNCVALUE.
131715                                    *AAX NUMPA;LDATX       % WRITEBACK MASK
131717                                    IF A< GO RSDIO         % MASK FOR DV10=100000
131721                                    IF A-20=0 GO RSXMS     % MASK FOR XMSG=20
131723                                    *AAX NOBYT-NUMPA;LDATX % IT IS DVINST
131724                                    *AAX -NOBYT
131726                                    A=:L; GO RSLAD         % NO. OF BYTES IN L
131726                                RSDIO:                     % IT IS DV10
131730                                    *AAX 11NOC+1-NUMPA;LDATX % NO. CHARS RETURNED
131732                                    A=:L;*AAX -11NOC-1
131732                                RSLAD:                     % LOAD BUFFER ADDRESS
131734                                    *AAX ISTRA; LDDTX      % BUFFER ADR IN DV10 & DVINST
131735                                    *AAX -ISTRA
131751                                    GO RSSTR; *)FILL
131751                                RSXMS:                     % IT IS XMSG FUNC XFREA
131753                                    *AAX DP5-NUMPA;LDATX   % NO OF BYTES RETURNED
131754                                    A=:L
131757                                    *AAX X5BUF-DP5;LDDTX;AAX  X5BUF
131760                                RSSTR:                     % STORE LENGHT (L) AND ADR(AD)
131761                                    *AAX 26ADD;STDTX       % BUFFER ADDRESS
131764                                    L=:A;*AAX 26NRB-26ADD;STATX      % NO. OF BYTES
131764                                FI
131764                            FI
131764                        FI
```

```
131764                  FI
131764               FI
131764            FI; PSWWAIT; X:=SWMSG; T:=5MBBANK; *STATX XN5ST      % MARK SWAPPER FREE
131770    *'8N500 -5MUCP
"131770"           X:=MAILINK; SWPWAIT=:D
131773            DO WHILE -1><X
131776               *LDATX XN5ST
131777               IF A=D THEN CALL 5ACTSWAPPER; GO FAR NXTMSG FI
132003               *LDXTX XLNK2
132004            OD; X:=SWMSG; *AAX SWPIN; STZTX                       % 0=:SWMSG.SWPINFO
132010            GO FAR NXTMSG
132011    *'8N500 5MUCP
"132011"   *)FILL
132015
132015    INTEGER POINTER PNSCANN:=NSCANN
132016
132016    SWPAGE:                                          % DISK I/O
132016            T:=5MBBANK; A:=PSW1WAIT; *STATX XN5ST
132021    * 8N500 5MUCP
"132021"           CALL ACTXRT; GO FAR NXTMSG
132023
132023    INTEGER CSUPROC
132024    LPRSUSPEND;
132024            T:=5MBBANK; X:=SWMSG; *AAX SUSPP; LDXTX               % X:=X.SUSPPROC
132030            IF X=0 OR X>>=CMXACTPROC GO ESWPFATAL
132034            X:=ADRPROC(X)=:CSUPROC
132036            IF X.PSTAT BIT SLICE THEN
132041               X:=X.MESSBUFF; CALL CHEXQ; GO ISP2
132044               IF PNSCANN=0 THEN
132046                  CALL TER500; GO FAR 5EN500; MIN PNSCANN; *TRR CCLR
132052               FI; T:=5MBBANK; *LDDTX XLNK2                       % D=X.N5STATUS
132054               SUSPSTAT; *STATX XN5ST                             % SUSPSTAT=:X.N5STATUS
132056               A:=D=:CSUPROC.SUSPMESSSTAT; -20=:X.LPRCOUNT
132063            FI
132063    ISP2:   X:=SWMSG; T:=5MBBANK; *AAX SWPIN; STZTX               % 0=:X.SWPINFO
132067            X:=SWMSG; CALL OKMONICO
132071            GO FAR NXTMSG; *)FILL
132104
132104    INTEGER CANSWER,CSWINF
132106    LDATREADY;
132106            X:=SWMSG
132107            ANSWER=:CANSWER; PSW1WAIT; T:=5MBBANK; *STATX XN5ST   % PSWWAIT=:X.N5STATUS
132114            *AAX SWPIN; LDATX
132116            IF A=0 GO ESWPFATAL; A=:CSWINF
132120            T:="MONAD"+A; A:=60=:L:=SWMSG+"MONAD"=:D:=5MBBANK; X:=A; *MOVPP
132132    SUSP2:  X:=CSWINF; T:=5MBBANK; CANSWER; *STATX XN5ST          % CANSWER=:SWMSG.SWPINFO.N5STATUS
132136            CALL TER500; GO FAR 5EN500
132140            CALL RRTWT;  GO FAR NXTMSG
132142
132142    LALLOPAGE:
132142            X:=SWMSG; PSW1WAIT; T:=5MBBANK; *STATX XN5ST           % PSWWAIT=:X.N5STATUS
132146            *AAX SWPIN; LDXTX
132150            IF X=0 GO ESWPFATAL; X=:CSWINF; *LDATX XMICF          % A:=SWMSG.SWPINFO.MICFUNC
132153            IF A=3SWMESS THEN EALLOPAGE ELSE ANSWER FI; A=:CANSWER
132162            X:="STOPREASON"; T:=CSWINF+X; A:=120=:L:=SWMSG+X=:D:=5MBBANK; X:=A; *MOVPP
132175            GO SUSP2
132176
132176    ESWPFATAL:  X:=SWMSG; SWPFATAL; CALL HRSTARTALL; GO FAR NXTMSG
132202
132202    *)FILL
```

```
132214    * 8N500 5MUCP
 132214     (CLTSB:
132214    * 8N500 -5MUCP
 132214           GO ESWPFATAL
132215    * 8N500 5MUCP
 132215
132215    *)FILL
132215
132215    % ***************************************************************************
132215    %       N 5 0 0 C
132215    % ***************************************************************************
132215
132215    %       MONITOR LEVEL ROUTINE TO EXECUTE COMMANDS
132215    %
132215    %       X-REG:                  PROCESS DESCRIPTION
132215
132215    INTEGER MMESSAGE=?,CCPROC=?
132215    INTEGER CCN5ST
132216    N5000C: X.CPUDF=:B; X=:CCPROC
132221            X:=X.T5BUFF=:MMESSAGE; T:=5MBBANK; *LDATX XN5ST      % X:=X.TBUFF.N5STATUS
132225            A=:CCN5ST
132226            IF A/\160000><0 THEN                                 % SOMETHING ABOUT POWER FAIL
132230               IF A BIT POWDET THEN
132232                  *IOF
132233                  C5STAT BONE BHPFAIL=:C5STAT
132236                  CCN5ST/\17777; *STATX XN5ST                    % A=:X.N5STATUS
132241                  CALL ITO500XQ; KPOWDOWN; CALL RSTARTALL; *ION
132245                  GO FAR RESENT
132246               ELSE
132247                  IF A BIT POWUP AND C5STAT NBIT BHPFAIL GO FAR RESENT
132254                  CALL ITO500XQ; TTMR=:TMR; GO FAR STUENT
132260               FI
132260            FI
132260            IF CCN5ST><MSGN500 AND A><WAITING GO FAR TOQUEUE
132267            IF FERROR><0 GO FAR XEILSTAT
132272            T:=5MBBANK; *LDATX XMICF
132274            IF A>>=MICFMAX GO FAR MILLFU
132277    MICR
132277            A GOSW
132277
132277               FAR PSTUPR,RMICVE,FAR MILLFU,FAR MILLFU,FAR MILLFU,
132305               SWMESS,EXAMD, DEPMD, RMEMD, WMEMD,
132312               CACHE, RAMED, WAMED,FAR RNEWCO,EXARG,
132317               DEPRG, REREG, WRREG, FAR MILLFU, FAR STAOPP,
132324               FAR MONCO,FAR MTRACO, STOP,  FAR MILLFU,MPHSREAD,
132331               MPHSWRITE,EXAMP, DEPMP, RMEMP, WMEMP,
132336               FAR FQUEUE, RAMEP, WAMEP, RLIMI, PRTRAP,
132343               WLIMI,FAR MILLFU;
132345    MICR;
132345    *)FILL
132366
132366    INTEGER MMESSAGE,CCPROC
132370
132370
132370    %       N O R D - 5 0 0   F U N C T I O N S
132370
132370    RMICVE:;WAMED:;RAMED:;EXARG:;DEPRG:;REREG:;RLIMI:;WLIMI:
132370    WRREG:; STOP:; EXAMP:;DEPMP:;RMEMP:;WMEMP:;PRTRAP:
132370    WAMEP:; RAMEP:;EXAMD:;DEPMD:;RMEMD:;WMEMD:;CACHE:
132370    MPHSREAD:; MPHSWRITE:
```

```
132370
132370          IF TMR << TTMR THEN A:=T FI
132375          GO FAR XTRACO
132376    *)FILL
132377
132377    SWMESS: IF INITFLAG BIT BRESPLACE GO FAR EEIRESPL
132402          T:=5MBBANK; *LDATX XSWFU                              % A:=X.SWFUNC
132404          IF A=MSWSTART THEN
132407             T:=5MBBANK; SWPPING; *STATX XN5ST                  % A=:X.N5STATUS
132412             A:=T; D:=X; X:=SWMSG; *AAX HSWPI; STDTX; AAX -HSWPI  % MMESSAGE=:SWMSG.SWPINFO
132420             3START; *STATX XMICF
132422             MSGN500; *STATX XN5ST; STZTX X5SND; STZTX X5REC
132426             SWACTIVE; *AAX SWPFU; STATX
132431             X:=ADRPROC(0); X.PSTAT BZERO SLICE=:X.PSTAT
132436             300=:X.5PRIORITY
132440    *)8N500 5MUCP
132440              *IOF
132441             X:=SWMSG; CALL TER500; GO FAR N500ERR; CALL ITO500XQ
132445    SWME1:   CALL LOWACT500; CALL FAR PIONPIOF; LTTMR=:TMR; *ION
132452             X:=MMESSAGE; ANSWER; GO FAR XEILSTAT
132455          FI
132455          IF A=MSWRSTART THEN
132460             X:=SWMSG; T:=5MBBANK; *AAX SWPIN; LDATX      % A:=SWMSG.SWPINFO
132464             IF A=MMESSAGE THEN
132467                T:=5MBBANK; *STZTX                         % 0=:SWMSG.SWPINFO
132471                X:=MMESSAGE; *AAX SWRST; LDATX             % A:=MMESSAGE.SWRSTAT
132474                T:=A; A:=0=:D; X:=SWMSG
132500                *IOF
132501                CALL MONICO
132502                CALL TER500; GO FAR N500ERR; GO SWME1
132505             FI; X:=MMESSAGE; GO FAR EIL5STAT; *)FILL
132524    INTEGER POINTER PMMESSAGE:=MMESSAGE,PCCPROC:=CCPROC
132526    INTEGER CCKFLIP
132527          FI
132527          IF A=MSWSWAIT THEN
132532             X:=SWMSG; T:=5MBBANK; *AAX SWPIN; LDATX      % A:=SWMSG.SWPINFO
132536             IF A><X:=PMMESSAGE GO FAR EIL5STAT
132541             *IOF
132542             CALL TER500; GO N500ERR; T:=5MBBANK; *AAX SKFLI; LDATX
132547             A=:CCKFLIP; SWPPING; X:=PMMESSAGE; *STATX XN5ST
132553             CALL ITO500XQ; T:=5MBBANK; *AAX SWRST; LDATX
132557             T:=CCKFLIP; A=:D:=0; X:=SWMSG; CALL MONICO
132564             X:=PMMESSAGE; T:=5MBBANK
132566             *AAX OLDMI; LDATX; AAX -OLDMI; STATX XMICF
132572             IF CCKFLIP=0 THEN
132574                L:=0; T:=5MBBANK
132576                *AAX PRET2; LDATX
132600                IF A><0 THEN
132601                   L BONE 1; *AAX RETP2-PRET2; LDDTX
132604                   X:=SWMSG; *AAX RETP2; STDTX
132607                FI; X:=PMMESSAGE; *AAX PRET4; LDATX
132612                IF A><0 THEN
132613                   L BONE 3; *AAX RETP4-PRET4; LDDTX
132616                   X:=SWMSG; *AAX RETP4; STDTX
132621                FI; X:=PMMESSAGE; *AAX PRET5; LDATX
132624                IF A><0 THEN
132625                   L BONE 4; *AAX RETP5-PRET5; LDDTX
132630                   X:=SWMSG; *AAX RETP5; STDTX
132633                FI; X:=PMMESSAGE; *AAX PRET6; LDATX
132636                IF A><0 THEN
```

```
132637                    L BONE 5; *AAX RETP6-PRET6; LDDTX
132642                    X:=SWMSG; *AAX RETP6; STDTX
132645                 FI; X:=SWMSG; A:=L; *AAX NUMPA; STATX
132651              FI
132651              CALL LOWACT500; CALL FAR PIONPIOF; *ION
132654              GO FAR PMONEN
132655           FI
132655           *IOF
132656           CALL 5ACTSWAPPER; GO TRACO
132660  *)FILL
132672
132672  MONCO:    X=:D; T:=5MBBANK; *AAX XADPR; LDXTX
132676            IF X.PSTAT BIT 5LTSLPRI AND A BIT SLICE THEN
132703               IF X.HTSLLOWPRI>=X.5PRIORITY THEN T+1=:X.5PRIORITY FI
132711            FI; X:=D
132712  STAOPP:
132712  MTRACO:
132712  TRACO:    LTTMR
132713  XTRACO:   A=:TMR
132714  TOQUEUE:  *IOF
132715           CALL TER500; GO N500ERR; CALL ITO500XQ; GO FAR MLOWACT
132721  *)FILL
132725  INTEGER ERSVX                    % SAVE FOR X WHILE DOING RSTARTALL
132726  N500ERR:  *IOF
132727           T:=5MBBANK; *STATX XN5ST
132731           X=:ERSVX
132732           CALL RSTARTALL; *ION
132734           X:=ERSVX
132735           GO FAR PMONEN
132736  EIRESPL:  EIRESPL; GO XEILSTAT
132740  MILLFU:   ILMICFUNC; GO XEILSTAT                    % ILLEGAL MICROPROGRAM FUNCTION
132742  EIL5STAT: ILN5STATUS                                % ILLEGAL N500 ANSWER
132743  XEILSTAT: T:=5MBBANK; *STATX XN5ST
132745  RESENT:   IF X><-1 THEN
132750               T:=5MBBANK; *LDXTX X5SND
132752               IF X<<CMXACTPROC THEN
132755                  X:=ADRPROC(X); X.MESSBUFF=:X.F5BUFF; 0=:X.T5BUFF
132761               FI
132761            FI
132761  RESENT:   GO FAR PMONEN
132762
132762  RNEWCO:   *IOF
132763           CALL TER500; 0/\0; CALL IFM500XQ
132766           CALL ACTRT; *ION
132770           X:=MAILINK; T:=5MBBANK
132772           DO WHILE -1><X
132775              X=:MESSLINK; *LDDTX X5SND
132777              IF A+1><0 AND A:=D*5SEMSIZE+"S500S".PSTAT BIT 5REWA THEN
133007                 *IOF
133010                 CALL IFM500XQ; *ION
133012                 GO FAR PSTUPR
133013              FI; X:=MESSLINK; *LDXTX XLNK2
133015           OD; GO FAR PSTUPR
133017
133017  INTEGER CCMESS
133020  FQUEUE:   *IOF
133021           X=:CCMESS:=SWMSG; CALL TER500; 0/\0; CALL IFM500XQ; 0/\0
133027           ANSWER; X:=CCMESS; T:=5MBBANK; *STATX XN5ST; ION
133034           GO RESENT
133035  *)FILL
```

```
133054
133054   %       I T O 5 0 0 X Q
133054   %
133054   % INSERT MESSAGE IN THE N500-EXECUTION QUEUE
133054   %
133054   % IF CALLED FROM OTHER LEVELS THAN DRIVER LEVEL THEN
133054   % ITO500XQ MUST BE CALLED IN "PIOF"
133054   %
133054   % ENTRY:      X=MESSAGE TO INSERT IN QUEUE
133054   %             B=N500 CPU-DATAFIELD
133054   INTEGER CMESS                  % CURRENT MESSAGE ADDRESS
133055   INTEGER PREVT,PREVX            % ADDRESS OF PREVIOUS MESSAGE IN QUEUE
133057   INTEGER POINTER RETXL
133060
133060   XITO500X: X=:CMESS; T:=L=:"RETXL":=5MBBANK; GO ITO5F
133065   ITO500XQ: X=:CMESS; T:=L=:"RETXL":=5MBBANK; *LDATX X5SND
133072             IF A+1=0 THEN                          % SPECIAL MESSAGE (HISTOGRAM ..)
133074                X:="MAILLINK"+B=:PREVX; 0=:PREVT; GO INS1
133101             FI; *AAX SPFLA; STZTX
133103            X:=CMESS
133104   ITO5F:     *AAX XADPR; LDXTX
133106             X.5PRIORITY=:L
133110             X:="MAILLINK"+B; T:=0; A:=-1
133114             DO
133114                X=:PREVX; T=:PREVT; *LDXTX XLNK2
133117             WHILE X><A
133121                X=:D; T:=5MBBANK; *AAX XADPR; LDXTX
133125                IF X:=X.5PRIORITY<L GO INSERT
133130                X:=D
133131             OD
133132   INSERT:   X:=PREVX
133133   INS1:     T:=PREVT; *LDDTX XLNK1
133135             X:=CMESS; T:=5MBBANK; *STDTX XLNK1
133140             T:=PREVT; X=:D:=PREVX; 5MBBANK; *STDTX XLNK1
133145             X:=CMESS; GO RETXL
133147
133147   %       T 5 T M Q U
133147   %
133147   % INSERT MESSAGE IN ND-500 TIME QUEUE
133147   %
133147   % ENTRY:      X=MESSAGE TO INSERT
133147   %             AD=TIME IN BASIC TIME UNITS WHEN THE PROC SHOULD BE STARTED
133147   %
133147   INTEGER CTM1,CTM2; DOUBLE CCTM=CTM1
133151   T5TMQU: T:=L=:"RETXL"; D-; *COPY CM1 ADC SA DA
133155           AD=:CCTM; X=:CMESS
133157           "N500DF"+"TIMLINK"=:X; T:=0; A:=-1
133164           DO
133164              X=:PREVX; T=:PREVT; *LDXTX XLNK2
133167           WHILE X><A
133171              X=:L; T:=5MBBANK; *AAX D5TIM; LDDTX
133175              A:=:D; A+CTM2; A:=:D; A:=A+C+CTM1
133202              IF A>=0 GO INSERT
133203              X:=L
133204           OD; GO INSERT
133206
133206
133206   %       I F M 5 0 0 X Q
133206   %
133206   % REMOVE MESSAGE FROM THE N500 EXECUTION QUEUE
```

```
133206    %
133206    % IF CALLED FROM LOWER LEVELS THAN THE DRIVER LEVEL THEN
133206    % IFM500XQ MUST BE CALLED IN "PIOF"
133206    %
133206    % ENTRY:      X=MESSAGE TO REMOVE
133206    %             B=N500 CPU-DATAFIELD
133206    %
133206    IFM500XQ: "MAILLINK"+8
133210    FQUS:   X=:D=:CMESS; T:=0
133213            DO
133213               T=:PREVT; X:=A; *LDATX XLNK2
133216            WHILE A><D                          % SEARCH EXECUTION QUEUE FOR OUR MSG
133220               IF A=-1 GO IFEX                  % END OF QUEUE? EXIT
133223               T:=5MBBANK
133224            OD; X=:PREVX:=A; T:=5MBBANK; *LDDTX XLNK1
133231            X:=PREVX; T:=PREVT; *STDTX XLNK1
133234    IFEX:   X:=CMESS; EXIT
133236
133236    %              F R 5 T M Q U
133236    %
133236    % REMOVE MESSAGE FROM ND-500 TIME QUEUE
133236    %
133236    % ENTRY:      X=MESSAGE TO REMOVE
133236    %
133236    FR5TMQU: "TIMLINK"+"N500DF"; GO FQUS
133241    *)FILL
133245
133245    %       I A C T 5 0 0
133245    %
133245    % ACTIVATE THE N500 COMMUNICATION
133245    %
133245    % CALLABLE FROM DRIVER LEVEL ONLY
133245    % IACT500 MUST BE CALLED IN "PIOF"
133245    %
133245    % ENTRY:      B=N500 CPU-DATAFIELD
133245    %             THE N500 IS ACTIVATED WITH THE START OF THE EXECUTION QUEUE
133245    %
133245    INTEGER POINTER LRAC
133246    INTEGER ACTX
133247    IACT500: IF C5STAT NBIT BHPFAIL THEN
133252               T:=HDEV+RSTA; *IOXT
133255               IF A NBIT 5CLOST THEN
133257                  X=:ACTX
133260                  IF A BIT 5ILOCK THEN L=:D; CALL TER500; 0/\0; L:=D FI
133266                  X:=MAILINK; T:=5MBBANK
133270                  DO
133270                     IF -1=X GO IACT2
133273                     *LDATX XN5ST
133274                     IF A=MSGN500 OR A=WAITING GO IACT1
133302                     T:=5MBBANK; *LDXTX XLNK2
133304                  OD
133305    IACT1:        A:=5MBBANK; T:=HDEV+LMAR;  *IOXT
133311                  A:=X; *IOXT
133313                  A:=5; T+"LCON-LMAR"; *IOXT
133316                  GO IACTF
133317
133317    IACT2:        X:=MAILINK; T:=5MBBANK
133321                  DO
133321                     IF -1=X GO IACT3
133324                     *LDDTX XN5ST
```

```
133325                      IF A=SUSPSTAT THEN
133330                         X:=:D; X:=ADRPROC(X); 0=:X.LPRCOUNT
133333                         X.SUSPMESSTAT; X:=D; T:=5MBBANK; *STATX XN5ST
133337                         GO IACT1
133340                      FI; T:=5MBBANK; *LDXTX XLNK2
133342                   OD
133343   IACT3:          A:=10; T:=HDEV+LCON;   *IOXT
133347                   A:=0;  T+"LSTA-LCON"; *IOXT
133352                   A:=1;  T+"LCON-LSTA"; *IOXT
133355                          T+"SLOC-LCON"; *IOXT
133357                   TTMR=:TMR
133361   IACTF:          X:=ACTX
133362                FI
133362             FI; EXIT
133363
133363
133363             INTEGER                SVXRG
133364
133364   %          I E R 5 0 0
133364   %
133364   % ROUTINE TO INTERRUPT N500
133364   %
133364   % WHEN CALLED FROM OTHER LEVELS THAN THE DRIVER LEVEL THEN
133364   % TER500 MUST BE CALLED IN "PIOF"
133364   %
133364   % ENTRY:       X=-1 THEN NO ACTUAL MESSAGE
133364   %              X><-1 THEN X=ACTUAL MESSAGE
133364   %
133364   % EXIT:        ERROR, N500 NOT INTERRUPTED
133364   %
133364   % EXIT+1:      OK, N500 STOPPED
133364   %
133364
133364   INTEGER POINTER TERLREG
133365   TER500: T:=HDEV+RSTA; *IOXT
133370             IF A BIT 5ILOCK THEN X=:SVXRG
133373               T+"TERMI-RSTA"; *IOXT
133375               T+"RSTA-TERMI"
133376               FOR X:=-10000 DO; *IOXT
133400               WHILE A BIT 5ILOCK
133402               OD; X:=SVXRG
133404             FI
133404             IF A BIT 5ILOCK THEN
133406                IF X >< -1 THEN
133411                   ESPTIMOUT; T:=5MBBANK; *STATX XN5ST
133414                FI; ESPTIMOUT=:FERROR
133416                A:=L=:"TERLREG"; CALL X5MCST; ESPTIMOUT; GO TERLREG
133423             FI; EXITA
133424   *)FILL
133432
133432   %          R R T W T
133432   %
133432   % WHEN CALLED FROM OTHER LEVELS THAN THE DRIVER LEVEL
133432   % THEN RRTWT MUST BE CALLED IN "PIOF"
133432   %
133432   % ROUTINE TO REMOVE MESSAGE FROM EXECUTION QUEUE
133432   % AND RESTART SINTRAN III PROGRAM
133432   %
133432   % ENTRY:       X=ACTUAL MESSAGE
133432   %
```

```
33432
33432   INTEGER POINTER RET4,RET10
33434   RRTWT: A:=L=:"RET4"
33436          IF X=SWMSG THEN
33441             T:=5MBBANK; *LDATX X5SND
33443             IF A=0 THEN EPFINSWAP; CALL RSTARTALL; GO RET4 FI
33447          FI; CALL IFM500XQ; K:="0"
33451   ACTRT: IF X><-1 THEN
33454             T:=5MBBANK; *LDATX X5SND
33456             IF A<<CMXACTPROC THEN
33461                X:=:A
33462   INXACT:      X:=ADRPROC(X); A=:X.F5BUFF; 0=:X.T5BUFF
33465                X:=:B
33466                IF K THEN CALL XRTACT ELSE CALL RTACT FI
33473                X:=:B:=X.F5BUFF
33475             FI
33475          FI; IF K GO RET10; GO RET4
33500
33500
133500  %       X A C T R T
133500  %       M U S T  BE CALLED WITH INTERUPT  O F F !
33500
133500  XACTRT: A:=L=:"RET10"; K:=1; GO ACTRT
133504  ACTXRT: A:=L=:"RET4"; A:=SWMSG; X:=0; K:="0"; GO INXACT
133512  *)FILL
133524
133524
133524  %       5 A C T S W A P P E R
133524  %
133524  % SUBROUTINE TO ACTIVATE THE N500 SWAPPER PROCESS
133524  %
133524  % WHEN CALLED FROM OTHER LEVELS THAN THE DRIVER LEVEL
133524  % THEN 5ACTSWAPPER MUST BE CALLED IN "PIOF"
133524  %
133524  % ENTRY:      X=MESSAGE REQUIREING SERVICE FROM SWAPPER
133524  %
133524  INTEGER POINTER RETAC
133525  INTEGER HMSGTOSW
133526  INTEGER MSGTOSW; DOUBLE CMSGTOSW=HMSGTOSW
133527  5ACTSWAPPER: A:=L=:"RETAC"
133531          T:=5MBBANK=:HMSGTOSW; X=:MSGTOSW; SWPWAIT; *STATX XN5ST
133536  * 8N500 5MUCP
133536           X:=SWMSG; *LDDTX XLNK2; AAX SWPFU; LDATX            % A:=X.SWPFUNC; D:=X.N5STATUS
133542  %      %IF D=PSWWAIT AND A=NEWSWAP  THEN
133542           IF D=PSWWAIT THEN                                  % SWAPPER FREE?
133545              X:=SWMSG; CALL OKMONICO
133547              T:=5MBBANK; A:=6; *AAX NUMPA; STATX
133553              AD:=CMSGTOSW; *AAX HSWPI-NUMPA; STDTX
133556              SWACTIVE; *AAX SWPFU-HSWPI; STATX
133561              X:=MSGTOSW; SWPPING; *STATX XN5ST
133564              CALL OFFESCAPE
133565              *LDDTX X5SIZ                                   % D:=X.MICFUNC
133566              IF 3SWMESS=D THEN
133571                 *LDATX XSWFU
133572              ELSE
133573                 *AAX TRAPN; LDATX
133575                 A=:D/\377; *STATX
133600                 A:=D SHZ -10
133602              FI; X:=SWMSG; *AAX SWPST; STATX
133605  * 8N500 5MUCP
```

```
"133605           ELSE
133606    *"8N500 5MUCP
"133606           FI
133606           X:=MSGTOSW; GO RETAC
133610    *)FILL
133615
133615
133615    %       R S T A R T A L L
133615    %
133615    % ROUTINE TO RESTART ALL SINTRAN III PROGRAMS WITH
133615    % MESSAGES IN THE N500 EXECUTION QUEUE WITH AN ERROR MESSAGE
133615    %
133615    % WHEN CALLED FROM OTHER LEVELS THAN THE DRIVER LEVEL
133615    % THEN RSTARTALL MUST BE CALLED IN "PIOF"
133615    %
133615    % ENTRY:      A=ERROR STATUS
133615    %
133615
133615    INTEGER POINTER CLRET
133616    INTEGER POINTER RET6
133617    INTEGER ERRSTAT
133620
133620    % HRSTARTALL IS ONLY CALLED FROM THE DRIVER LEVEL
133620    HRSTARTALL: T:=L=:"CLRET"; *IOF
133623                CALL RSTARTALL; *ION
133625                GO CLRET
133626
133626    RSTARTALL: A=:ERRSTAT; A:=L=:"RET6"
133631           X:=MAILINK; T:=5MBBANK
133633           DO WHILE -1><X
133636              IF ERRSTAT BIT POWDOWN THEN
133641                 IF SWMSG=X GO RSTA1
133644                 *LDATX XN5ST
133645                 A BONE POWDOWN
133646              ELSE
133647                 A=:FERROR
133650              FI; *STATX XN5ST
133651              CALL IFM500XQ
133652              IF X><SWMSG THEN CALL FAR XACTRT FI
133656    RSTA1:    T:=5MBBANK; *LDXTX XLNK2
133660           OD; GO RET6
133662    *)FILL
133667
133667    N5FUD:
133667    *"8N500 8F5UD
"133667    %-------------------------------------------------------------------
133667    % ACTIVATED WHEN MON 333 (UDMA) FROM ND500,   DRIVER LEVEL
133667    % CALLED FROM MCHANDEL
133667    % ENTRY:    B= ND-500 CPU DATAFIELD
133667    %           X= MONITOR CALL MESSAGE IN 5MBBANK
133667    % RETURN:
133667    %   EXIT    ERROR, TRY AGAIN ON LEVEL 1
133667    %   EXIT+1 OK, UDRxx RT-PROG STARTED
133667    % -----------------------------------  -----------------------------
133667    %
133667
133667    % FIXED 500 MEMORY DESCRIPTION
133667    % TX DISP
133667    SYMBOL XFFPA=10              % FIRST LOGICAL PAGE
133667    SYMBOL XFPHP=30              % ND-100 PHYS ADDR
```

```
133667    SYMBOL FIXCONT=14,FIXABST
133667
133667    INTEGER POINTER LREG=?
133667    INTEGER N5CP=?,N5ME=?,RTPD=?,ALOP=?
133667    INTEGER N5SE=?,UFU=?,UPIO=?,UNI=?,CCAP=?
133667    DOUBLE  DBUA=?,IPAR1=?
133667
133667
133667    DISP 7                          % SPECIAL DF. FOR UDRXX RT-PROG
133667          INTEGER UR5M                          % ADDRESS TO MESSAGE FROM N500 PROC.
133667          INTEGER POINTER FUNCP                 % FUNCTION
133667          DOUBLE  POINTER MEMOP                 % MEMORY ADDRESS
133667          INTEGER POINTER BLOCP                 % NOT USED
133667          DOUBLE  POINTER NWORP                 % NR OF 16 BITS WORDS
133667    PSID
133667
133667          X=:N5ME;                              % SAVE ADDRESS TO CURRENT MESSAGE AND 500 DATAFIELD
133670          A:=L=:"LREG"
133672    %     ------------------------ GET PARAMETERS
133672          T:=5MBBANK; *LDATX X5SND              % GET 500 PROC. NO
133674          A=:N5SE
133675          *AAX 5PPA2; LDDTX                     % GET BUFFER ADDRESS
133677          AD=:DBUA;   *AAX AP1-5PPA2; LDDTX     % GET FUNCTION
133702          IF A >< 0 GO UERR; A:=D=:UFU
133705          *AAX AP2-AP1; LDDTX                   % GET PIO DATA
133707          A:=D=:UPIO
133711          *AAX AP3-AP2; LDDTX                   % GET LOG DEV
133713          IF A >< 0 GO UERR; A:=D=:UNI
133716          *AAX AP4-AP3; LDDTX                   % GET IPAR1
133720          AD=:IPAR1
133721
133721          IF UNI < LUDV OR > HUDV GO UERR       % CHECK LOGICAL UNIT
133730          CALL LOGPH  IF A=0 GO UERR; D=:A      %
133733          IF A=0 GO UERR;
133734          T:=5MBBANK; X:=N5ME;* AAX XADPR; LDXTX
133740          IF X.RTRES >< D.RTRES GO UERR         % DEVICE NOT RESERVED
133745
133745          X:=X.N5RDF=:RTPD                      % ADDRESS TO SPECIAL DF. FOR UDRXX
133747          X=:B; X:=N5ME                         % B=SPECIAL DF. X=MESSAGE
133751          IF RTRES = 0 GO UERR                  % UDRxx (RT-PROG) NOT STARTED
133753     %  ------------------------ FUNCTION CODE
133753          UFU; CALL UDFUNCT; A=:FUNCP
133756    %     ------------------------ BUFFER ADDRESS  & LENGTH
133756          IF A >=0 AND A <=3 THEN               % DOES FUNCTION USE BUFFER?
133762              CALL COAF;                        % BUFFER FIXED ?, CONV. ADDR
133763              AD=:MEMOP                         % ND 100 PHYS. ADDR.
133764              IF UFU =0 OR =2 THEN              % INPUT FUNCTION?
133771                  IF CCAP NBIT 17 GO UERR       % CHECK FOR WRITE PERMIT ON SEGMENT
133774              FI
133774          ELSE
133775    %     ------------------------ DIV. IN IPAR1
133775            AD:=IPAR1                           %
133776            IF A >= 54 AND <= 57 THEN           % PIO FUNCTION
134004               A:=UPIO                          % PIO DATA
134005               A:=:D                            % A = US CONT LINES, D = PIODATA
134006            FI
134006            AD=:NWORP                           % WRITE IPAR1 IN ABST PAR. LIST
134007        FI
134007    %     ------------------------ START UDRXX PROG
134[ ]          X:=N5ME=:UR5M                         % MESSAGE ADDRESS
```

```
134011          T:=5M8BANK;
134012          20; * STATX XN5ST                        % SET 500 PROC WAITING
134014          CALL RTACT
134015          MIN "LREG";                              % OK SKIP RETURN
134016  UERR:   X:=N5ME
134017          CALL XSETCPU                             % RESTORE B-REG TO CPU DF
134020          GO LREG
134021
134021  INTEGER POINTER LREG
134022  INTEGER N5CP,N5ME,RTPD,ALOP
134026  INTEGER N5SE,UFU,UPIO,UNI,CCAP
134033  DOUBLE  DBUA,IPAR1
134037  INTEGER CPYS,FLP,LLP,CDIS,MXTA
134044  *)FILL
134054  %-------------------------------------------------------------
134054  % LOCAL SUBROUTINE, TEST LEGAL FUNCTION CODE FOR MON UDMA
134054  %     A = FUNCTION CODE
134054  UDFUNCT:
134054  @ICR;   IF A=0 OR =1 OR =2 OR =3 OR =7     % LEGAL FUNCTION CODE?
134054               OR =20 OR =21 OR =24 OR =54
134054               OR =55 OR =56 OR =57 OR =62
134054               OR =64 OR =65 OR =70 THEN
134054  @CR;           EXIT                              % OK, RETURN
134133               ELSE
134134                 GO UERR                           % PARAMETER 1 IS ILLEGAL
134135          FI
134135  *)FILL
134135
134135  %-------------------------------------------------------------
134135  % LOCAL SUBROUTINE, CHECK IF BUFFER IN 500-PROC IS FIXED CONT.
134135  %
134135
134135  COAF:   AD:=IPAR1                               % LENGTH
134136          IF D BIT "0" GO UERR                     % ODD BYTE COUNT
134140          AD SHZ -1 =:NWORP                       % CONVERT TO NUMBER OF ND-100 WORDS
134142          AD SHZ 6;A=:ALOP                        % NUMBER OF LOGICAL PAGES -1
134144          AD:=DBUA                                % ND-500 BUFFER ADDRESS
134145          IF D BIT "0" GO UERR                     % ODD BYTE ADDRESS?
134147          AD SHZ -13;  A=:D                       % D = LOGICAL SEG NO
134151          A:=N5SE-1*MFANT+5DSPS+D=:X
134157          T:=5FXBNK; *LDATX                       % A = CAPABILITY
134161          A=:CCAP                                 % DATA SEGMENT CAPABILITY
134162          A /\ 1777 =: CPYS                       % CURRENT PHYSICAL SEGMENT
134164          AD:=DBUA SHZ 5;                         % A= PAGE NO
134166          D SHZ -6                                % D = DIP (ND-100)
134167          A=:FLP + ALOP =:LLP                     % 1. AND LAST LOG PAG
134172          A:=D=:CDIS                              % DISPLACEMENT WITHIN PAGE
134174          A:=N5SE-1*"MFSIZE*MFANT"+5FXTBL=:X+"MFSIZE*MFANT"=:MXTA
134203          DO
134203              T:=5FXBNK; *LDATX                   % A = MFDESCR
134205              IF A BIT FIXABS OR A BIT FIXCONT THEN
134211              IF A /\ 1777 = CPYS THEN
134215                  T:=5FXBNK; *LDDTX XFFPA             % A=FIRST LOG PAGE, D = LAST
134217                  IF A <= FLP AND D >= LLP THEN       %%  FIXED AREA FOUND
134225                      T:=5FXBNK; *LDATX XFPHP         %%  ND-100 PHYS PAGE
134227                      AD SHZ -20; AD SHZ 12;          %%  A:=D; A=0; AD=PAGE ADDR
134231                      T:=CDIS; D+T; A:=A+C                %%  ND 100 PHYS ADDR IN AD
134234                      EXIT
134235                  FI
134235              FI;FI
```

```
134235              X:=X+MFSIZE
134236            WHILE X<< MXTA
134241            OD
134242            GO FAR UERR; *)FILL
134252    *)KILL YENDC; YENDC=*
134252    *7ENDC/
074255
074255    % MONITOR FUNCTION, REMOVING RT-PROG FROM I/O WAIT
074255    RIOWA: X=:B; IF X:=RTRES=0 GO MONEN
074261           X.STATUS BZERO 5WAIT=:X.STATUS
074264           GO STUPR
074265    *)FILL
074267    *)KILL 7ENDC; 7ENDC=*
074267    *YENDC/
134252    *'8N500 -8F5UD
"134252    *)KILL YENDC; YENDC=*
134252    *7ENDC/
074267    *'8N500 8F5UD
 074267    %=============================================================-=====
074267    %              5 D T U P
074267    %
074267    % ROUTINE CALLED FROM THE SYSTEM MONITOR TO UPDATE A COPY
074267    % OF THE DATA-CAPABILITY TABLE AND THE FIXED-INFO TABLE
074267    % THESE TABLES ARE USED IN THE FAST MON UDMA INTERFACE
074267    %
074267    % ENTRY:       X=ND-500 PROCESS NUMBER
074267    %              A=ADDRESS OF FIX-INFO TABLE OR DATA-CAPABILITY TABLE ON THE
074267    %                SYSTEM MONITOR'S DATA SEGMENT
074267    %              T=FLAG; =1: COPY FIX-INFO TABLE
074267    %                      =2: COPY DATA CAPABILITY TABLE
074267    %
074267    % EXIT:        ONLY B-REGISTER IS SAVED
074267    %
074267    INTEGER 5DMPY
074270    *'8N500
"074270    5DTUP:
074270    *'8N500 8F5UD
"074270             *PIOF
074271           IF T-1=0 THEN T:="MFSIZE*MFANT"=:5DMPY:=5FXTBL; GO LABLZ FI  % FIX-INFO TABLE
074300           IF T-1=0 THEN T:=MFNCP=:5DMPY:=5DSPS; GO LABLZ FI            % DATA CAPABILITY
074307           *PION
074310           GO 5DTU9                                          % ERROR
074311    LABLZ: A=:D;=X-1*5DMPY; T+A                              % T=DESTINATION ADDR
074316           X:=5FXBNK; A:=5DMPY:=:L; *PION; MOVNP             % COPY TABLE
074323           A=:L
074324    *'8N500
"074324    5DTU9: EXIT
074325    *)FILL
074331    *)KILL 7ENDC; 7ENDC=*
074331    *YENDC/
134252
134252    % ===========================================================================
134252    %       M C H A N D E L
134252    %
134252    % DECODING MONITOR CALL MESSAGES FROM N500
134252    %
134252    % CALLED FROM : DECOMESS
134252    %
134252    % DRIVER LEVEL
134252    %
```

```
134252    % ENTRY:      X=ACTUAL MESSAGE (=MESSLINK)
134252    %             THE ACTUAL PROCESS IS FOUND IN PROCAD
134252    %
134252
134252    INTEGER POINTER RET7=?
134252    MCHANDEL: A:=L=:"RET7"
134254            IF 5MLOPROC><0 THEN                                    % MONITOR CALL LOGING?
134256               T:=5MBBANK; *LDATX X5SND
134260               IF A=5MLOPROC OR A:=-1=T THEN
134266                  T:=5MBBANK; *AAX MCNO; LDATX
134271                  IF A<<1000 THEN
134274                     A SH 1; T:="N500DF".BUADR; X:=X.5BUADR+A; *LDDTX  %INCREMENT  -
134302                     D+1; A:=A+C; *STDTX                          % MONITOR CALL COUNT
134305                     X:=MESSLINK
134306                  FI
134306               FI
134306            FI; T:=5MBBANK; *AAX XADPR; LDATX
134311            A=:PROCAD
134312            *AAX MCNO-XADPR; LDATX; AAX SMCNO-MCNO; STATX
134316            IF A=2TUSED THEN                                       % FAST MONITOR-CALL HANDELING
134321               AD:=PROCAD.500TUSED; T:=0; X:=MESSLINK; CALL MONICO
134326               GO RET7
134327            FI; X:=MESSLINK
134330            IF A=N5SWAP THEN                                       % SPEC. MONITOR CALL FROM SWAPPER
134333               IF X=SWMSG THEN
134336                  CALL FAR SWPDECODER
134337               ELSE
134340                  IF 5DBFLAG><0 THEN E5DEBUG; CALL HRSTARTALL; GO FAR NXTMSG FI
134345                  T:=5MBBANK; 25; *STATX XN5ST                     % YOU ARE NOT AUTHORIZE TO DO THIS
134350                  CALL RRTWT                                       % WILL BE CONVERTED TO ILL. MON CALL
134351               FI; GO RET7
134352            FI
134352    *"8N500 8F5UD
"134352              IF A=333 THEN CALL FAR N5FUD; GO NORMMC; GO RET7 FI % MON UDMA, MAY BE FAST CALL
134360    *"8N500
"134360             IF A >= L12MIN AND A <= L12MAX THEN                  % SPECIAL HANDLING ON LEVEL 12?
134366               T:=INMCALL=:PROCAD.5RUNSTATUS; X:=MESSLINK
134372    @ICR
134372               A-L12MIN GOSW STAPROC,NSTOPROC,SWITPROC,FAR NINSTR,FAR NOUTSTR,
134401               FAR GERRC, FAR 5SIBMO, FAR SPRIO, FAR SWMC, FAR DVIO,
134406               FAR A5XMSG, FAR B5XMSG,M5TMOUT;
134411    @CR;
134411            FI
134411
134411    NORMMC: CALL RRTWT; GO RET7
134413
134413    INTEGER POINTER RET7
134414    *)FILL
134447
134447    %
134447    %       SPECIAL TREATMENT OF MONITOR CALLS ON LEVEL 12
134447    %
134447
134447    NSTOPROC:
134447            IF PROCAD.CDTSEGN BIT 55REP THEN
134453               A BZERO 55REP=:X.CDTSEGN; X:=MESSLINK; GO PRST
134457            FI; STOPPED; T:=5MBBANK; X:=MESSLINK; *STATX XN5ST
134463            GO RET7
134464
134464    STAPROC:
```

```
134464    SWITPROC:
134464            T:=5MBBANK; *AAX NPROC; LDDTX
134467            IF A = 0 OR A >> CMXACTPROC GO ILPROC
134473            X:=ADRPROC(A)
134475            IF X.MAGNO >< D OR X.RTRES = 0 THEN
134502    ILPROC:    A:=EILPROC
134503    TOEMONCO:  X:=MESSLINK; CALL EMONICO
134505             GO RET7
134506            FI; X=:D
134507    %  - IF PROCESS STOPPED (BY STOPPR ETC.) THEN RESTART IT
134507            IF X:=X.T5BUF><0 THEN
134511               T:=5MBBANK; *LDATX XN5ST
134513               IF A=STOPPED THEN
134516                  CALL OKMONICO
134517                  GO TSTSW
134520               ELSE IF A=I5TMQ THEN                          % IN TIME QUEUE
134524                  CALL FR5TMQU; CALL TER500; 0/\0
134527                  A:=1; CALL SPITMQ; CALL IT0500XQ           % START PROC
134532                  GO TSTSW
134533               FI; FI
134533            FI
134533    % - PROCESS CURRENTLY NOT STOPPED
134533            D.CDTSEGN BONE 55REP=:X.CDTSEGN                  % SET REPEAT FLAG
134537    TSTSW:
134537            X:=MESSLINK; T:=5MBBANK; *AAX MCNO; LDATX
134543            IF A=PRSWITCH GO NSTOPROC
134546            *AAX -MCNO
134547    PRST:   CALL OKMONICO
134550    TRET7:  GO RET7
134551            *)FILL
134565
134565    %
134565    % ND-500 TMOUT MONITOR CALL
134565    %
134565    INTEGER CRREASON(0)
134565    DOUBLE CC5ATM
134567    M5TMOUT:  IF PROCAD.CDTSEG BIT 55REP THEN
134573                 A BZERO 55REP=:X.CDTSEGN; A:=-1
134576    N5TMF:       A=:CRREASON
134577               X:=MESSLINK; CALL TER500; 0/\0
134602               A:=CRREASON; CALL SPITMQ; GO TRET7
134605             FI; X:=MESSLINK; T:=5MBBANK; *AAX ADP1; LDDTX      % AD=NO. OF TIME UNITS
134611             IF A><0 GO LILP1; IF D=0 GO N5TMF                  % RESTART WITH RESTART REASON=0 WHEN ZERO NUMBE
134614             D=:L; *AAX ADP2-ADP1; LDDTX                        % AD=TIME UNIT
134617             IF A><0 OR D<=0 OR D>4 GO LILP1
134625             A:=D-1 SH 1; BCLCN(A); T:=L; 0=:X=:L
134635             DO WHILE T><0
134637                 IF T BIT 0 THEN L+D; X+A+C FI                  % COMPUTE TIME IN BASIC TIME UNITS
134643                 T SHZ -1; AD SH 1
134645             OD; D:=X; A:=L; A+"N500DF".5ATM2; A:=:D; A:=A+C+X.5ATM1; AD=:CC5ATM
134656             X:=MESSLINK; T:=5MBBANK; *AAX D5TM1
134661             CC5ATM; *STDTX
134663             X:=MESSLINK; CALL IFM500XQ                         % SAVE TIME IN MESSAGE
134665             I5TMQU; T:=5MBBANK; *STATX XN5ST
134670             CALL FAR T5TMQU; GO TRET7
134672
134672    D:IO:
134672    NOOUTSTR:
134672            CALL GTDF; GO FAR NORMMC
134674            A:=D; X:=MESSLINK; T:=5MBBANK; *AAX TODF; STATX
```

```
134701          *AAX DNOBY-TODF; LDDTX
134703          IF A >< 0 OR D >> 4000 THEN
134707  EILP1:          X:=MESSLINK
134710  EILPAR:         A:=EILPAR; GO FAR TOEMONCO
134712          FI
134712          IF MIFLAG NBIT WSMC THEN
134715                  X:=MESSLINK; T:=5MBBANK; 3RMED; *STATX XMICF
134721                  MSGN500; *STATX XN5ST
134723                  A:=D; *AAX NRBYT; STATX
134726                  *AAX OSTRA-NRBYT; LDDTX; AAX N500A-OSTRA; STDTX
134732                  *AAX ABUFA-N500A; LDDTX; AAX N100A-ABUFA; STDTX
134736                  "STTDRIV"; *AAX SPFLA-N100A; STATX
134741                  GO TRET7; *)FILL
134757          FI
134757          *AAX -DNOBY
134760
134760  %       S T T D R I V
134760  %
134760  %       START TERMINAL DRIVER.    CALLED WHEN THE LOGICAL DATA READ
134760  %       IS FINISHED IN DIRECT OUTSTRING TO TERMINAL
134760  %       X=ND-500 MESSAGE ADDR
134760  STTDRIV:
134760          T:=5MBBANK; *AAX TODF; LDATX
134763          A:=B; *AAX LBUFA-TODF; LDATX; AAX 5HENT-LBUFA; STATX
134770          *AAX SPFLA-5HENT; STZTX; AAX -SPFLA
134773          X=:A:=:B; T:="ON5MSG"; CALL XSTDFADDR; X:=:B
135000          N5IOWAIT; T:=5MBBANK; *STATX XN5ST
135003          "L3STDV"=:MFUNC; CALL RTACT
135006          T:=5MBBANK; *AAX XADPR; LDXTX
135011          X:=X.CPUDF=:B; GO FAR NXTMSG
135014  *)FILL
135022  *)KILL YENDC; YENDC=*
135022  */ENDC/
074331
074331  % OUTSTRING RESTART - ND-500 DRIVER LEVEL (LEVEL 12)
074331  % ENTRY:      X=ND-500 MESSAGE ADDR
074331
074331  OSTRS: *POF
074332          CALL XSETCPU; T:=5MBBANK; *AAX SMCNO; LDATX
074336          IF A=511 THEN                          % DVIO
074341             T:=5MBBANK
074342             *AAX XADPR-SMCNO; LDATX             % PROCAD MUST BE RESTORED
074344             A=:PROCAD
074345             *AAX 11DMA-XADPR; LDDTX             % LOAD MAX NUMBER OF BYTES AND CONINUE
074347             *AAX -11DMA                         % IN INSTRING
074350             X=:MESSLINK; GO FAR XNINSTR
074352          FI; *AAX -SMCNO
074353          CALL OKMONICO; CALL IACT500; GO FAR CALLID12
074356
074356  %
074356  %       T 5 R S T
074356  %       RESTART ND-500 FROM THE TERMINAL OUTPUT DRIVER  (LEVEL 10)
074356  %       ENTRY:     X = ND-500 PROCESS MESSAGE
074356  %
074356  T5RST:  A:=X; *IRW LV12B DX
074360          "OSTRS"; *IRW LV12B DP
074362          LV12; *MST PID; PION; PIOF
074366          EXIT
074367
074367  % START DRIVER  -  MONITOR LEVEL
```

```
074367    % ENTRY: X=TERMINAL DATAFIELD
074367
074367    INTEGER POINTER CPITO:=177000+5BFPAGE+5BFPAGE
074370    L3STDV: X=:B; *PIOF
074372            IF TYPRING BIT 5TERM THEN
074375               CALL SPTOWINDOW; *PON
074377               CALL STDEV; *POF
074401               0=:CPITO
074402            ELSE
074403               CALL STDEV
074404            FI; *PION
074405            GO MONEN
074406    *)FILL
074422    *)KILL 7ENDC; 7ENDC=*
074422    *YENDC/
135022
135022
135022    %
135022    %       M O V T A B
135022    %
135022    % MOVE BREAK/ECHO TABLE FROM THE ND-500 MESSAGE TO THE TERMINAL DATAFEIELD
135022    %
135022    % ENTRY:       X=ND-500 MESSAGE
135022    %              T=DISPLACEMENT IN THE DATAFIELD TO MOVE INTO
135022    %              A=DISPLACEMENT IN THE ND-500 MESSAGE TO MOVE FROM
135022    %
135022    INTEGER POINTER MOVLR
135023    INTEGER CTFLGADDR,CSVT
135025    MOVTAB: A+X=:D; T=:CSVT:=5MBBANK; *AAX TODF; LDXTX
135033            T:=CSVT
135034            IF X.TYPRING BIT 5TERM THEN
135037               X:=X.TDFLGADDR; T+X; X:=TDFBANK
135042            ELSE
135043               T+X; X:=0
135045            FI; A:=10:=:L=:"MOVLR":=5MBBANK; *MOVPP
135052            GO MOVLR
135053    *)FILL
135055
135055    %
135055    %       N I N S T R
135055    %
135055    NINSTR:  T:=5MBBANK; X:=MESSLINK; *AAX DMAXB; LDDTX; AAX -DMAXB
135062    ^NINSTR: IF A >< 0 OR D >> 4000 GO LILP1
135066            CALL GTDF; GO FAR NORMMC
135070            A=:X
135071            T:="FLAGB"                          % TERMINAL INPUT DF
135072            CALL XGTDFADDR
135073            IF A BIT 5LSTA THEN                 % GET FLAGB INTO A
135075                  T:="IN5MSG"                   % HAVE WE LOST OUR TERMINAL?
135076                  CALL XGTDFADDR
135077                  A=:L:=0                       % GET ADDR OF ND-500 MSG
135101                  CALL XSTDFADDR                % SAVE IT AND CLEAR IT IN DF
135102                  L=:X
135103                  A:=316                        % USED BY EMONICO
135104                  CALL EMONICO                  % TERMINAL LINE IS NOT CONNECTED
135105                  GO FAR NXTMSG                 % GIVE USER THE SAD INFO
135106            FI
135106            X=:A
135107            X:=MESSLINK; T:=5MBBANK; *AAX TODF; STATX
135113            IF A.TYPRING BIT 5TERM THEN X.TDFLGADDR/\1777+"5UBFPAGE*2000"=:X FI
```

```
135123          X=:CTFLGADDR:=MESSLINK; *AAX 5BRST; LDATX
135127          IF A=-1 THEN A:=0; GO NOBRK FI
135134          IF A<<11 THEN
135137                IF A = 7 THEN                          % USER SUPPLIED BREAK STRATEGY
135142                   A:="5BRKTAB"; T:="PBRK7"            % DISPLACEMENT IN MSG AND DF
135144                   X:=MESSLINK; CALL MOVTAB
135146                   A:=10                               % CONTINUE AS IF REUSE OLD
135147                FI
135147                IF A=10 THEN                           % USE OLD BREAK STRATEGY?
135152                   A:="PBRK7"+CTFLGADDR
135154                ELSE
135155                   BRKTB(A)
135157                FI
135157   NOBRK:       X:=MESSLINK; T:=5MBBANK; *AAX TODF; LDXTX
135163                T:="BRKTAB"; CALL XSTDFADDR
135165          FI; T:=5MBBANK; X:=MESSLINK; *AAX 5ECHS; LDATX
135171          IF A=-1 THEN A:=0; GO NECHO FI
135176          IF A<<11 THEN
135201                IF A = 7 THEN                          % USER SUPPLIED ECHO STRATEGY
135204                   A:="5ECHTAB"; T:="PECH7"
135206                   X:=MESSLINK; CALL MOVTAB; A:=10
135211                FI
135211                IF A=10 THEN A:="PECH7"+CTFLGADDR ELSE ECHTB(A) FI
135221                GO NECHO; *)FILL
135246   NECHO:       X:=MESSLINK; T:=5MBBANK; *AAX TODF; LDXTX
135252                T:="ECHOTAB"; CALL XSTDFADDR
135254          FI; N5IOWAIT; X:=MESSLINK; T:=5MBBANK; *STATX XN5ST
135260          *AAX 5FYLL; STZTX; AAX MLFLA-5FYLL; STZTX; AAX SPFLA-MLFLA; STZTX
135266   % PUT MAX NUMBER OF CHARACTERS INTO BRKMAX
135266          *AAX SMCNO-SPFLA; LDATX
135270          A-511
135271          IF A = 0 THEN
135272                *AAX 11DMA+1-SMCNO; LDATX    % DVIO
135274                *AAX TODF-11DMA-1
135275          ELSE
135276                *AAX MAXBY-SMCNO; LDATX    % DVINST
135300                *AAX TODF-MAXBY
135301          FI
135301          *LDXTX                                 % TODF
135302          T:="BRKMAX"; CALL XSTDFADDR
135304   % PUT ADDR OF THIS MSG INTO TERMINAL DF (USED BY TERNMINAL INPUT DRIVER)
135304          MESSLINK; T:="IN5MSG"; CALL XSTDFADDR
135307          T:="BHOLD"; CALL XGTDFADDR
135311          IF A>0 THEN
135313                X:=:B:=X.MESSLINK; T:=5MBBANK; A:=1; *AAX MLFLA; STATX
135321                "IBMOVE"=:MFUNC; CALL RTACT
135324          ELSE
135325                IF X.ISTAT = -1 OR = -2 THEN
135334                   A:=0; T:="IN5MSG"; CALL XSTDFADDR
135337                   X:=MESSLINK; T:=5MBBANK; *AAX SMCNO; LDATX
135343                   IF A = 511 THEN
135346                      A:=0=:D; T:=5MBBANK; *AAX 11NOC-SMCNO; STDTX
135353                      100000; *AAX NUMPA-11NOC; STATX
135356                   ELSE
135357                      A:=0=:D; T:=5MBBANK; *AAX NOCHR-SMCNO; STDTX
135364                      4; *AAX NUMPA-NOCHR; STATX
135367                   FI; A:=3; X:=MESSLINK; CALL EMONICO
135372                   T:=5MBBANK; *AAX TODF; LDXTX
135375                FI; T:="DFLAG"; CALL XGTDFADDR
135377                A BONE 5ECHO; T:="DFLAG"; CALL XSTDFADDR
```

```
135404          FI; PROCAD.CPUDF=:B; GO FAR NXTMSG
135406          *)FILL
135425
135425  %
135425  %       G E R R C
135425  %       GET ERROR CODE.  USED AFTER A PROGRAMMED TRAP.
135425  %       DRIVER LEVEL
135425  %
135425  INTEGER CNTXBANK
135426  GERRC:  T:=5MBBANK; *LDATX X5REC
135430          A+1*REGBSZ+"ERREG"=:T; "N500DF".CNTXPAGE+X.ADRZERO+X.NPPGUWIP=:D:=0
135442          AD SHZ 12; D+T; A:=A+C; D=:X=:L; A=:T=:CNTXBANK; *LDDTX
135452          X:=MESSLINK; T:=5MBBANK; *AAX ERRCO; STDTX
135456          A:=0=:D; X:=L; T:=CNTXBANK; *STDTX
135463          X:=MESSLINK; CALL OKMONICO
135465          T:=5MBBANK; A:=1; *AAX NUMPA; STATX
135471          GO FAR NXTMSG
135472
135472  %
135472  %       5 S I B M O
135472  %       SPECIAL MONITOR CALL FROM SIBAS SERVER IN ND-500
135472  %
135472  INTEGER CSIBNO
135473  INTEGER POINTER PRET7:=RET7
135474  5SIBMO: T:=5MBBANK; X:=MESSLINK; *AAX SIBNO; LDDTX
135500          IF A><0 OR D>>"MXSIBAS" OR T=0 GO FAR LILP1
135506          X:=D=:CSIBNO:=SIBBDEVS(X)=:D
135512          IF X.RTRES><PROCAD.RTRES THEN X:=MESSLINK; 5; CALL EMONICO; PRET7=:P FI
135524          1=:D.SIB500; X:=SIBAPDEVS(CSIBNO)=:D; 0=:X.SRTCSTAT; X=:CSIBNO
135534          IF X:=X.RTRES><0 AND X.STATUS BIT 5WAIT THEN
135541             D:=:B; CALL RTACT; B:=D
135544          FI; X:=MESSLINK; T:=5MBBANK; *AAX SIBST; LDDTX
135550          IF A=0 AND D=0 GO FAR NSTOPROC
135553          1=:CSIBNO.SRTCSTAT; X:=MESSLINK; CALL OKMONICO; PRET7=:P
135562
135562  *)FILL
135600
135600  %       S P R I O
135600  %       SET PRIORITY
135600  SPRIO:  IF PROCAD.CDTSEGN BIT SPROK THEN
135604                X:=MESSLINK; T:=5MBBANK; *AAX NEWPR; LDATX
135610                IF A=0 THEN
135611                   *AAX -NEWPR; LDXTX X5SND
135613                   5TSLSTATUS(X) BONE 5BRKF=:5TSLSTATUS(X)
135616                   PROCAD.PSTAT BONE SLICE=:X.PSTAT; 20=:X.5PRIORITY
135624                ELSE
135625                   IF A >>= 300 THEN A:=EILPAR; GO SPERET FI
135632                   A=:PROCAD.5PRIORITY; X.PSTAT BZERO SLICE=:X.PSTAT
135637                FI
135637                X:=MESSLINK; CALL TER500; GO FAR TOERET
135642                CALL IFM500XQ; CALL ITO500XQ; CALL OKMONICO
135645          ELSE
135646                A:=EPRMC
135647  SPERET:       T:=5MBBANK; X:=MESSLINK; CALL EMONICO
135652          FI; GO FAR NXTMSG
135653  *)FILL
135667
135667
135667  %       S W M C
135667  %       MONITOR CALL TO THE SWAPPER
```

```
135667   SWMC:  MSM510 SHZ 10=:D; T:=5MBBANK; *AAX TRAPN; LDATX
135675          A+D; *STATX; AAX -TRAPN
135700          CALL 5ACTSWAPPER; GO FAR NXTMSG
135702   *)FILL
135705
135705
135705
135705   %       T O E R E T
135705   %
135705   %       CALL HRSTARALL ON DRIVER LEVEL
135705   %
135705   TOERET: CALL HRSTARTALL; GO FAR CALLID12
135707
135707
135707   %
135707   %       G E T   T E R M I N A L   D A T A   F I E L D
135707   %
135707   % ENTRY:       X=ND-500 MESSAGE ADDR
135707   %
135707   % EXIT:
135707   %
135707   % EXIT+1:      A=ADDR OF TERMINAL INPUT DATAFIELD
135707   %              D=ADDR OF TERMINAL OUTPUT DATAFIELD
135707   %
135707   INTEGER POINTER GTDFRET
135710   GTDF:  A:=L=:"GTDFRET"
135712          T:=5MBBANK; *AAX DOUTD; LDDTX; AAX -DOUTD
135716          IF A><0 GO GTDFRET
135720          IF D=1 AND PROCAD.OUTDF><0 GO NRET
135726          X:=MESSLINK; T:=5MBBANK; *AAX OUTDE; LDATX; AAX -OUTDE
135733          CALL LOGPH; IF D = 0 GO GTDFRET
135736          A:=PROCAD.RTRES=:X+"BRESLINK"; X:=:A
135743          DO WHILE X >< D AND X >< A
135747               X:=X.RESLINK
135750          OD
135751          IF X = D THEN              % THE TERMINAL IS RESERVED
135753               IF X.TYPRING BIT 5TERM THEN A:=X; GO NRET FI
135760          FI
135760   NLRET: X:=MESSLINK; GO GTDFRET
135762   NRET:  T:="XDFOPP"; X:=A; CALL XGTDFADDR; X=:D
135766          MIN "GTDFRET"; GO GTDFRET
135770
135770          *)FILL
136000
136000   %
136000   %       P I B M O V E                    MONITOR LEVEL
136000   %       MOVES BYTES FROM TERMINAL INPUT BUFFER TO ND-500 BUFFER
136000   %       IF MAX NO OF BYTES IS REACHED THE THE ND-500 PROCESS IS RESTARTED.
136000   %       GOES TO PMONEN IF THE INPUT BUFFER IS EMPTY.
136000   %
136000   % ENTRY:       X=TERMINAL DATAFIELD
136000   %
136000
136000   *)KILL YENDC; YENDC=*
136000   *7ENDC/
074422
074422   INTEGER CCIN5MSG=?
074422   IBMOVE: X=:B
074423          *PIOF
074424          CALL SPT3WINDOW
```

```
074425          *PION
074426          IF FLAGB BIT 5LSTA THEN                 % TERMINAL LINE LOST?
074431                X:=IN5MSG
074432                0=:IN5MSG
074433                A:=316                            % GODBYE TERMINAL
074434                *POF
074435                CALL EMONICO
074436                GO MONEN
074437          FI
074437          IN5MSG=:CCIN5MSG
074441  IIBM:   *PION; IOF
074443  IIBMOVE:
074443          CALL IOTRANS; GO TMWT; *POF                     % READ CHARACTER
074446          CALL FAR 5BPUT                                  % SAVE IN BUFFER
074447          T:=5MBBANK; X:=CCIN5MSG; *AAX 5FYLL; LDATX; AAA 1; STATX  % INCR. 5FYLLE
074455          A=:D                                  % SAVE 5FYLLE FOR LATER TEST
074456          *AAX SMCNO-5FYLL; LDATX               % MONITOR CALL NUMBER
074460          A-511
074461          IF A=0 THEN                           % DVIO OR DVINST?
074462                *AAX 11DMA-SMCNO+1              % DVIO
074463          ELSE
074464                *AAX MAXBY-SMCNO
074465          FI
074465          *LDATX                                % MAX NUMBER OF BYTES
074466          IF D>=A GO N5RST                      % BREAK REACHED? (5FYLLE>=MAX NO. BYTES)
074470          *PON
074471          IF T:=RSISTE>=0 THEN
074474             IF HENTE=T GO N5RST
074477          ELSE
074500             IF BRECHOFL BIT 5BREAK GO N5RST
074503          FI; GO IIBM
074504  *)FILL
074513  INTEGER CCIN5MSG
074514
074514  N5RST:  *POF
074515          X:=CCIN5MSG; T:=5MBBANK; *AAX 5FYLL-1; LDDTX
074521          IF D = 0 THEN CALL ERRFATAL FI
074524          *AAX SMCNO+1-5FYLL; LDATX
074526          IF A = 511 THEN
074531             *AAX 11NOC-SMCNO
074532          ELSE
074533             *AAX NOCHR-SMCNO
074534          FI; T:=5MBBANK; A:=0; *STDTX
074537          X:=CCIN5MSG; *LDXTX X5SND
074541          5TSLSTATUS(X) BONE 5BRKF=:5TSLSTATUS(X)
074544          *PON
074545          0=:IN5MSG; *POF
074547          X:=CCIN5MSG; CALL XSETCPU; T:=5MBBANK
074552          IF MIFLAG BIT WSMC THEN
074555                3WMONCO; *STATX XMICF
074557                *AAX ISTRA; LDDTX; AAX 26ADD-ISTRA; STDTX
074563                *AAX 5FYLL-26ADD; LDATX; AAX 26NRB-5FYLL; STATX
074567                *AAX KFLIP-26NRB; STZTX
074571                A:=0=:D; *AAX FUNCV-KFLIP; STDTX; AAX -FUNCV  % 0=:FUNCVALUE
074576               %%A:=4; *AAX NUMPA-FUNCV; STATX; AAX -NUMPA
074576                CALL DVMASK
074577                CALL MCCO                                     % SET WRITEBACK MASK
074600          ELSE
074601                3WMED; *STATX XMICF
074603                MSGN500; *STATX XN5ST
```

```
074605                  *AAX ABUFA; LDDTX; AAX N100A-ABUFA; STDTX
074611                  *AAX ISTRA-N100A; LDDTX; AAX N500A-ISTRA; STDTX
074615                  *AAX 5FYLL-N500A; LDATX; AAX NRBYT-5FYLL; STATX
074621                  "INSMONCO"; *AAX SPFLA-NRBYT; STATX
074624          FI; X:=CCIN5MSG
074625          CALL RPIT3
074626          CALL TER500; GO FAR N500ERR; GO FAR MLOWACT
074631  TMWT:   *PON
074632          IF TDRADDR.ISTAT=-1 OR A=-2 GO N5RST
074642          *PIOF
074643          X:=CCIN5MSG; T:=5MBBANK; *AAX MLFLA; STZTX
074647          CALL RPIT3
074650          *ION
074651          GO FAR PMONEN
074652          *)FILL
074667
074667
074667  %
074667  %       5 B P U T
074667  %       PUT ONE BYTE INTO 2K BYTES BUFFER OF A ND-500 PROCESS
074667  %       B = INPUT TERMINAL DATA FIELD.
074667  %
074667  INTEGER C5BYT
074670  5BPUT:  A=:C5BYT; X:=CCIN5MSG; T:=5MBBANK
074673          *AAX HBUFA; LDDTX                               % A:=HBUFA; D:=LBUFA
074675          *AAX 5FYLL-HBUFA; LDXTX                         % X:=5FYLL
074677          *BLDA 0 DX
074700          T:=X SHZ -1; D+T=:X; T:=A
074705          IF K THEN
074707             *LDATX
074710             A\/C5BYT
074711          ELSE
074712             C5BYT SHZ 10
074714          FI; *STATX
074715          EXIT
074716  *)FILL
074717  *)KILL 7ENDC; 7ENDC=*
074717  *ENDC/
136000
136000  %
136000  %       INSMONCO RESTARTS THE ND-500 PROCESS AFTER THE INPUT BYTES
136000  %       ARE WRITTEN TO THE DATA MEMORY IN THE QUICK INSTRING MONITOR CALL 504.
136000
136000  INSMONCO: T:=5MBBANK; *AAX SPFLA; STZTX; AAX -SPFLA     % CLEAR SPFLAG
136004            CALL OKMONICO
136005            CALL DVMASK                    % SET WRITEBACK MASK
136006            GO FAR NXTMSG
136007
136007  % DVMASK SETS THE WRITE-BACK MASK IN THE MESSAGE AFTER MON DVINST
136007  % OR MON DVIO. CALLED FROM INSMONCO AND N5RST
136007  % X=MESSAGE
136007  DVMASK:
136007            T:=5MBBANK
136010            *AAX SMCNO; LDATX              % MONITOR CALL NUMBER
136012            A-511                          % DVIO = MON 511
136013            IF A=0 THEN
136014               A:=100000                   % WRITE BACK MASK FOR DVIO
136015            ELSE
136016               A:=4                        % WRITE BACK MASK FOR DVINST
136017            FI
```

```
136017              *AAX NUMPA-SMCNO; STATX       % STORE WRITE BACK MASK
136021              *AAX -NUMPA
136022              EXIT
136023
136023    INTEGER PROCADD                         % ADDRESS OF PROCESS DESCRIPTION
136024    *)FILL
136032
136032
136032    %=============================================================================%
136032    %                                                                             %
136032    %       X M S G                                                               %
136032    %                                                                             %
136032    %       XMSG INTERFACE ROUTINES.  EXECUTED ON LEVEL 12                        %
136032    %                                                                             %
136032    %=============================================================================%
136032
136032    SYMBOL XFDUS=43                 % DEFINE USER SEGMENT AND PIT INFO
136032
136032    INTEGER ARRAY XTARR(5MXPROC+1)
136131    INTEGER EXTARR(0)
136131    *XTARR=EXTAR
136131    *)ZERO
136131
136131    @ICR
136131    INTEGER ARRAY XMRETMASK:=(
136131            0, 0, 4, 0,34, 0,20,20, 0,14, 2, 0, 0,74,74, 4,
136151            2, 0, 0, 0, 0, 0,14, 4, 0, 0, 0, 0, 0, 0, 0, 0,
136171            0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
136203    @CR;
136203    SYMBOL X5MASK=77                % MASK FOR OBTAINING XMSG FUNCTION NUMBER
136203    SYMBOL X5MAX=51                 % MAX XMSG FUNCTION NUMBER
136203    INTEGER TRXMSG=?
136203
136203    %       X M W K
136204    %       STARTED FROM XMSG WHEN A MESSAGE ARRIVES ON A PORT BELONGING
136203    %       TO ND-500 PROCESS.
136203    %       B = ND-500 MESSAGE
136203    XMWK:   T:=5MBBANK; X:=B; *AAX XADPR; LDXTX
136207            X=:B; CALL NW5ST; 0/\0
136212    XMWKR:  CPUDF=:B; GO FAR CALLID12
136215    *)FILL
136220
136220
136220    A5XMSG:
136220    B5XMSG;
136220            X=:B; T:=5MBBANK; *LDXTX X5SND
136223            IF XTARR(X) = 0 THEN
136225                  B=:X
136226                  *AAX HBUFA; LDATX
136230                  T:=XFDUS; L:=0; *MON 2XMSG
136233                  GO FAR X5LEAVE; IF T < 0 GO FAR X5ERET
136236                  X:=B; T:=5MBBANK; *LDXTX X5SND
136241                  A:=L=:XTARR(X); T:=XFWDF
136244                  "XMWK"; *MON 2XMSG
136246                  GO FAR X5LEAVE; T:=5MBBANK; X:=B; *LDXTX X5SND
136252                  A:=XTARR(X)
136253            FI; A=:L; X:=B; *AAX N5XFU; LDATX; AAX -N5XFU
136260            A=:TRXMSG
136263            IF A /\ X5MASK >> X5MAXF GO X5ILPAR
136265    @CR;
```

```
136265          A GOSW
136265          LFDUM, LFDCT, LFGET, LFREL, LFRHD, LFWHD, LFREA, LFWRI,
136276          LFSCM, LFMST, LFOPN, LFCLS, LFSND, LFRCV, LFPST, LFGST,
136306          LFSIN, LFSRL, LFABR, LFABW, LFMLK, LFMUL, LFM2P, LFP2M,
136316          LFRIN, LFCRD, LFSTD, LFDIB, LFRIB, LFWIB, LFPRV, LFRTN,
136326          LFRRH, LFDUB, LFWDF, LFDUS, LFSMC, LFDMM, LFALM, LFFRM,
136336          L5FUN, LFDPS;
136340  R;


136340  X5ILPAR:
136340  LFSRL:;LFABR:;LFABW:
136340  LFRIN:;LFCRD:;LFSTD:;LFDIB:;LFRIB:;LFWIB:;LFPRV:
136340  LFDUB:;LFWDF:;LFDUS:;L5FUN:;LFDPS:
136340          CALL SETCPU; GO FAR LILPAR
136342
136342  LFDCT:
136342  LFGET:
136342  LFREL:
136342  LFMST:
136342  LFOPN:
136342  LFCLS:
136342  LFRCV:
136342  LFPST:
136342  LFGST:
136342  LFRRH:
136342  LFSCM:
136342  LFRTN:
136342  LFRHD:
136342  LFWHD:
136342  LFP2M:
136342  LFDMM:
136342  LFFRM:
136342  LFALM:
136342          T:=5MBBANK; *AAX DP3-1; LDDTX; AAX DP2+1-DP3; LDATX; AAX DP4-DP2; LDXTX
136351          GO MONXM
136352  LFSMC:                          % USE B5XMSG
136352  LFREA:
136352  LFWRI:                          % USE B5XMSG
136352          T:=TRXMSG
136353          T BONE XFUSG=:TRXMSG:=5MBBANK
136356          *AAX DP2-1; LDDTX; AAX LBUFA+1-DP2; LDATX; AAX DP4-LBUFA; LDXTX
136364          GO MONXM
136365  LFSND:
136365  LFM2P:
136365          T:=5MBBANK; *AAX ADP2; LDDTX; AAX DP3-ADP2; LDXTX
136372
136372  LFDUM:; LFMLK:; LFMUL:; LFSIN:
136372  MONXM:  T:=TRXMSG; *MON 2XMSG           % STARTS AN XMSG FUNCTION
136374          GO X5LEAVE; GO X5RET
136376
136376  %       LEVEL 14 PART OF XMSG FINISHED.
136376  X5LEAVE:  X:=B; T:=5MBBANK
136400            CALL SETCPU
136401            TRXMSG-1                      % DISCONNECT FUNCTION?
136403            IF A><0 THEN
136404                 N5IOWAIT; *STATX XN5ST
136406            ELSE                          % FUNCTION WAS DISCONNECT
136407                 CALL OKMON               % RETURN TO USER IMEDIATELY
136410            FI
```

```
136410              GO FAR NXTMSG
136411
136411    INTEGER TRXMSG
136412    *)FILL                          % N5XFUNC
136425
136425
136425    %       XMSG FUNCTION FINISHED AND LEVEL 12 RESTARTED HERE
136425
136425    X5RET:  IF T < 0 GO FAR X5ERET
136427            X:=:B; T=:L:=5MBBANK; *AAX FUNCA; STZTX
136434            A:=:L; *AAX FUNCD-FUNCA; STATX
136437            *AAX N5XFU-FUNCD; LDATX; AAX -N5XFU
136442            A /\ X5MASK:=:L
136444    @ICR
136444            L GOSW
136444            RFDUM, RFDCT, RFGET, RFREL, RFRHD, RFWHD, RFREA, RFWRI,
136455            RFSCM, RFMST, RFOPN, RFCLS, RFSND, RFRCV, RFPST, RFGST,
136465            RFSIN, RFSRL, RFABR, RFABW, RFMLK, RFMUL, RFM2P, RFP2M,
136475            RFRIN, RFCRD, RFSTD, RFDIB, RFRIB, RFWIB, RFPRV, RFRTN,
136505            RFRRH, RFDUB, RFWDF, RFDUS, RFSMC, RFDMM, RFALM, RFFRM,
136515            R5FUN, RFDPS;
136517    @CR;
136517
136517    RFM2P:
136517    RFRCV:
136517    RFPST:
136517    RFGST:
136517    RFGET:
136517    RFRRH:
136517            *AAX AP3; STZTX; AAX DP3-AP3; STATX; RCLR DA; AAX ADP4-DP3; STDTX
136526            D:=B; *AAX ADP5-ADP4; STDTX
136531            A:=:D; *AAX FUNCD-ADP5; LDATX
136534            A:=:D; *AAX ADP6-FUNCD; STDTX; AAX -ADP6
136540            GO X5L1; *)FILL
136544    RFRHD:
136544            *AAX DP4-1; STDTX; AAX DP3+1-DP4; STATX
136550            A:=B; *AAX DP5-DP3; STATX
136553            *AAX AP3-DP5; STZTX; AAX AP4-AP3; STZTX; AAX  AP5-AP4; STZTX; AAX -AP5
136562            GO X5L1
136563    RFWHD:
136563    RFMST:
136563    RFP2M:
136563            *AAX ADP3; STDTX
136565            D:=B; A:=0; *AAX ADP4-ADP3; STDTX; AAX -ADP4
136572            GO X5L1
136573    RFOPN:
136573    RFCLS:
136573    RFDCT:
136573    RFREL:
136573    RFSND:
136573    RFRTN:
136573    RFSCM:
136573    RFSIN:
136573            A=:D:=0; *AAX ADP2; STDTX; AAX -ADP2
136600            GO X5L1
136601    RFSMC:
136601            DP4 SHZ 3               % NUMBER OF CALLS MULTIPLIED BY 8 (SIZE OF A CALL)
136603            A=:D                    % GIVES NUMBER OF BYTES TO RETURN
136604    RFREA:
136604            A:=0; *AAX ADP5; STDTX
```

```
136607          A:=D; *AAX 26NRB-ADP5; STATX
136612          *AAX X5BUF-26NRB; LDDTX; AAX 26ADD-X5BUF; STDTX; AAX -26ADD
136617          3WMONCO; *STATX XMICF
136621          GO X5L3
136622  RFALM:
136622  RFDMM:
136622  RFFRM:
136622  RFWRI:
136622          A:=0; *AAX ADP5; STDTX; AAX -ADP5
136626
136626  RFDUM:
136626  RFMLK:
136626  RFMUL:
136626  X5L1:   3MONCO; *STATX XMICF
136630  X5L3:   *AAX N5XFU; LDATX; AAX -N5XFU
136633          X=:B; A /\ X5MASK=:X; XMRETMASK(X); X:=B
136640          *AAX NUMPA; STATX; AAX KFLIP-NUMPA; STZTX; AAX -KFLIP
136645          CALL MCCO
136646  X5L2:   CALL XSETCPU; CALL IACT500; GO FAR CALLID12
136651  X5ERET: B=:X; T=:A:=1; AD SH -20; CALL MONICO; GO X5L2
136657
136657  RFSRL:;RFABR:;RFABW:
136657  RFRIN:;RFCRD:;RFSTD:;RFDIB:;RFRIB:;RFWIB:;RFPRV:
136657  RFDUB:;RFWDF:;RFDUS:;R5FUN:;RFDPS:
136657          CALL ERRFATAL
136660  *)FILL
136670  *)KILL YENDC; YENDC=*
136670  *7ENDC/
074717
074717  INTEGER POINTER XNW5RET:=NW5RET
074720  INTEGER POINTER IOLRG
074721  PNW5ST: IF X:=TDRADDR.RTRES=0 OR X.WLINK=0 THEN EXIT FI; *POF
074730          A:=L=:"IOLRG":="PNW5RET"=:XNW5RET
074734          GO FAR INW5ST
074735  PNW5RET: MIN "IOLRG"; *PON
074737          GO IOLRG
074740  INTEGER POINTER L10LR
074741  PT5RST: A:=:L=:"L10LR":=L; *POF
074745          CALL T5RST; *PON
074747          GO L10LR
074750  %
074750  %       X 5 D C N
074750  %       ROUTINE FOR DISCONNECT FROM XMSG.  ON LEVEL 1
074750  X5DCN:  A=:X-"S500S"=:D:=0; T:=5SEMSIZE; * RDIV ST; PIOF; COPY SX DD
074760          IF XTARR(A) >< 0 THEN
074763                *IRW LV12B DL
074764                0=:XTARR(X); D.CPUDF; *IRW LV12B DB
074770                "12DCN"; *IRW LV12B DP
074772                LV12; *MST PID
074774          FI; *PION; EXIT
074776
074776  12DCN:  *POF
075000          GO FAR P12DCN
075000
075000  *)FILL
075010  *)KILL 7ENDC; 7ENDC=*
075010  *7ENDC/
136670
136670  P12DCN: T:=XFDCT; * MON 2XMSG
136672          GO FAR CALLID12; GO FAR CALLID12
```

```
136674
136674
136674    %=============================================================================
136674    %       S E T C P U
136674    %       SET THE B REGISTER = THE CPU DATAFIELD OF THE CPU THE PROCESS IS RUNNING ON.
136674    %       ASSUMES B = ND-500 MESSAGE
136674    SETCPU:   X:=B
136675    XSETCPU:  X=:D; T:=5MBBANK; *LDXTX X5SND
136700              X:=ADRPROC(X); X.CPUDF=:B; X:=D; EXIT
136705
136705
136705    %=============================================================================
136705    %       NW5ST
136705    %       NO WAIT ND-500 START.
136705    %       CALLED ON BREAK WHEN TERMINAL OR INTERNAL DEVICE IS IN NO WAIT MODUS
136705    %       MUST BE CALLED IN IOF.
136705
136705    INTEGER NWBSAVE
136706    INTEGER POINTER NW5RET
136707    NW5ST: A:=L=:"NW5RET"; IF RTRES.WLINK = 0 GO NWRET
136714    INW5ST: D:=X; CALL FSEMA; GO NWRET; X=:D
136720            IF X:=X.T5BUF><0 THEN
136722               T:=5MBBANK; *LDATX XN5ST
136724               IF A=I5TMQU THEN                          % PROCESS IN TIME QUEUE
136727                  A:=B=:NWBSAV; CALL XSETCPU
136732                  CALL FR5TMQU; CALL TER500; 0/\0
136735                  A:=1; CALL SPITMQU; CALL IT0500XQ      % ACTIVATE PROCESS
136740                  GO INW5A
136741               FI
136741               IF A><STOPPED GO INW5X
136744               A:=B=:NWBSAVE; CALL OKMONICO; CALL XSETCPU
136750    INW5A:     CALL IACT500; NWBSAVE=:B
136753               T:=5MBBANK; *LDXTX X5SND
136755               5TSLSTATUS(X) BONE 5BRKF=:5TSLSTATUS(X)
136760            ELSE
136761    INW5X:        D.CDTSEGN BONE 55REP=:X.CDTSEGN
136765                  A:=X-"S500S"=:D:=0; T:=5SEMSIZE; *RDIV ST
136773                  5TSLSTATUS(A) BONE 5BRKF=:5TSLSTATUS(X)
136777            FI; MIN "NW5RET"
137000    NWRET: GO NW5RET
137001
137001    *)FILL
137016
137016    %=============================================================================
137016    %               S P I T M Q
137016    %
137016    % START PROCESS CURRENTLY IN TIME QUEUE
137016    %
137016    % ENTRY:        X=MESSAGE TO RESTART
137016    %               A=RESTART REASON
137016    %
137016    INTEGER POINTER SPIRET
137017    SPITMQ: T:=L=:"SPIRET"
137021            A=:D; IF A><-1 THEN A:=0 FI
137026            T:=5MBBANK; *AAX ADP3; STDTX; AAX -ADP3
137032            CALL OKMONICO; T:=5MBBANK
137034            A:=4; *AAX NUMPA; STATX; AAX -NUMPA
137040            GO SPIRET
137041
137041    %=============================================================================
```

```
137041   %       M O N C O
137041
137041   %       X-REG: MESSAGE ADDRESS
137041   %       AD-REG: FUNCTION VALUE
137041   %       T-REG: K FLIP-FLOP
137041
137041   INTEGER KKFLIP
137042   EMONICO:  A=:D:=0; T:=1; GO MONICO
137046   OKMONICO: T:=0; A:=0; D:=0
137051   MONICO:   T=:KKFLIP:=5MBBANK; *AAX FUNCV; STDTX
137055             A:=KKFLIP; *AAX KFLIP-FUNCV; STATX; AAX NUMPA-KFLIP; STZTX
137062             3MONCO; *AAX -NUMPA; STATX XMICF        % RESTART AFTER MONITOR CALL
137065   MONCO:    T:=5MBBANK; 140300; *AAX H500A; STATX; AAX -H500A
137072             X=:D; MSGN500; *STATX XN5ST; AAX XADPR; LDXTX
137077             ACTIVE=:X.5RUNSTATUS; X:=D; EXIT
137103
137103
137103   *)FILL
137107    L
137107   @ELIB
137107   @LIB TOD
137107   @LIB CXCPU
137107   *)FILL
137107
137107   %==========================================
137107   %       H I S T S A M P L E
137107   %
137107   % DRIVER LEVEL
137107   % X=HIMESS
137107
137107   INTEGER POINTER RET9
137110   INTEGER CAAT
137111   HISTSAMPLE: A:=L=:"RET9"
137113           T:=5MBBANK; *AAX WANTP-1; LDDTX; AAX ACPRO-WANTP+1; LDATX
137120           IF A=D THEN
137122              *AAX -ACPRO
137123              MFREE; *STATX XN5ST
137125              5HISTART; A=:CAAT; D=:L; *AAX N500A; LDDTX
137132              D-L; A:=A+C-1-CAAT
137136              IF A<0 GO OUTSIDE; T:=5HINTERVAL; Z:="0"; *RDIV ST
137142              IF A < 5HICHANNELS AND Z NBIT THEN
137147                 A SH 1=:X; 5HIDATA(X); D+1; A:=A+C; AD=:5HIDATA(X)
137155              ELSE
137156   OUTSIDE:      5HIOUTSIDE; D+1; A:=A+C; AD=:5HIOUTSIDE
137162           FI;FI; GO RET9
137163
137163   %       C L S T A T U S
137163   % ROUTINE TO CLEAR THE N500 INTERFACE STATUS
137163   %
137163   % WHEN CALLED FROM OTHER LEVELS THAN THE DRIVER LEVEL THEN
137163   % CLESTATUS MUST BE CALLED IN "PIOF"
137163   %
137163   % ENTRY:        A=MASK TO CLEAR STATUS WITH
137163   %
137163   % EXIT:         A=N500 STATUS AFTER CLEARING
137163   %
137163
137163   INTEGER RET8
137164
137164   CLESTATUS:
```

```
137164          A=:D; T:=HDEV+RSTA; *IOXT
137170          IF A BIT 5POWOF OR A BIT 5PFAIL THEN
137174             IF A BIT 5PFAIL OR C5STAT BIT BHPFAIL THEN A:=L=:RET8
137203                CALL TER500; GO CLABORT
137205                10;    T:=HDEV+LCON;  *IOXT
137211                       T+"RSTA-LCON"; *IOXT
137213                A/\D;  T+"LSTA-RSTA"; *IOXT
137216                "0";   T+"LCON-LSTA"; *IOXT
137221    CLABORT:    A:=RET8=:L; T:=HDEV+RSTA; *IOXT
137226             ELSE A BONE 5POWOF BZERO 5PFAIL
137231             FI
137231          FI; EXIT
137232
137232    *)FILL
137241
137241    % DISABLE ESCAPE WHILE USING SWAPPER
137241    %
137241    % ENTRY:      X=ND-500 MESSAGE
137241    INTEGER CSVX,CCSVT
137243    OFFESCAPE: X=:CSVX; T=:CCSVT:=5MBBANK; *AAX XADPR; LDXTX
137250               X.PSTAT; *BLDA 150 DA
137252               A BONE 5BRK=:X.PSTAT; X.CDTSEGN; *BSTA 150 DA
137256               A BZERO 52ESCSET=:X.CDTSEGN; X:=CSVX; T:=CCSVT; EXIT
137263
137263    %
137263    % ESCAPE ON AFTER USING SWAPPER
137263    %
137263    % ENTRY:      X=ND-500 MESSAGE
137263    %
137263    % EXIT:       A=ESCSTOP
137263    %             ESCAPE IS TYPED WHILE USING SWAPPER
137263    %
137263    % EXIT+1:     NO ESCAPE TYPED, CONTINUE
137263    %
137263    ONESCAPE:  X=:CSVX; T:=5MBBANK; *AAX XADPR; LDXTX
137267               X.PSTAT BZERO 5BRK; T:=X.CDTSEGN
137272               *BLDA 150 DT; BSTA 150 DA
137274               A=:X.PSTAT
137275               IF T BIT 52ESCSET AND A NBIT 5BRK THEN
137301                  T BZERO 52ESCSET; L-1; A BONE 5BRK=:X.PSTAT; ESCSTOP
137306               FI; T BZERO 5XBRK=:X.CDTSEGN
137310               X:=CSVX; T:=5MBBANK; EXITA
137313    *)FILL
137315
137315    %=============================================================================
137315    %              C H 5 C P U P R E S E N T
137315    %
137315    % SUBROUTINE CALLED FROM THE START-UP (SINTR) SEQUENCE TO CHECK IF
137315    % THE GENERATED ND-500 CPU'S EXISTS OR NOT
137315    %
137315    CH5CPUPRESENT: B=:D; A:="S5CPUDF"=:B:=0; *TRR IIE
137322           DO WHILE B<<="E5CPUDF"
137325              T:=HDEV+RSTA; *TRA IIC
137330              A:=200; *TRR IIE; IOXT; TRA IIC
137334              IF A=0 THEN 1=:CPUAVAILABLE FI
137337              A:=5CPUDFSIZE; B+A
137341           OD; D=:B; EXIT
137344    *)FILL
137346
137346    *)KILL YENDC; YENDC=*
```

```
137346   */ENDC/
075010
075010
075010   %       C H E X Q
075010   %
075010   % SUBROUTINE TO CHECK IF A ND-500 MESSAGE IS IN THE ND-500 EXECCUTION QUEUE
075010   %
075010   % ENTRY:       X = ND-500 MESSAGE
075010   %              B = ND-500 CPU DATAFIELD
075010   % EXIT:        THE MESSAGE IS NOT IN THE QUEUE
075010   % EXIT+1:      THE MESSAGE IS IN THE QUEUE
075010   %
075010   CHEXQ: X=:D; T:=5MBBANK; X:=MAILINK
075013          DO WHILE -1><X
075016             IF X=D THEN EXITA FI
075021             *LDXTX XLNK2
075022          OD; X:=D; EXIT
075025
075025
075025
075025   %       R T B A K
075025   %
075025   % MONITOR LEVEL ROUTINE TO REMOVE SINTRAN III PROGRAM FROM I/O-WAIT
075025   %
075025   % ENTRY:       X=PROCESS DESCRIPTION
075025   %
075025   RTBAK: X=:B
075026          IF X:=RTRES><0 THEN
075030             IF X.WLINK = 0 THEN
075032                CALL FTIMQU; CALL TOEXQ; GO INRTBAK
075035             FI
075035             IF PSTAT BIT 5REWA THEN
075040                A BZERO 5REWA=:PSTAT
075042   INRTBAK:     X.STATUS BZERO 5WAIT=:X.STATUS
075045             FI; 1=:MTOR
075047          FI; GO FAR MONEN
075050
075050   %
075050   % MONITOR LEVEL ROUTINE TO ACTIVATE ND-100 APPLICATION
075050   % WAITING FOR SIBAS IN ND-500
075050   %
075050   APLRSTART: X=:B; IF X:=RTRES><0 GO INRTBAK
075054              GO MONEN
075055   *)FILL
075062
075062   %       5 M C S T
075062   %
075062   % SUBROUTINE TO STOP N500 (MICRO STOP)
075062   %
075062   X5MCST: T:=HDEV
075063   5MCST: T+UNLC; *IOXT                        % UNLOCK
075065          A:=40; T+"LCON-UNLC"; *IOXT          % DISABLE TAG-IN DECODING
075070          A:=2;  T+"RETG-LCON"; *IOXT
075073          A:=0;  T+"5MCLR-RETG";*IOXT          % MASTER CLEAR
075076          EXIT
075077
075077   *)FILL
075077   *)KILL 7ENDC; 7ENDC=*
075077   */ENDC/
137346
```

```
137346
137346    INTEGER MXMPNT:=0                 % MAX BUFFER ADDRESS
137347    INTEGER FMPNT:=0                  % FIRST BUFFER ADDRESS
137350    INTEGER CMPNT:=0                  % CURRENT MESSAGE ADDRESS
137351    INTEGER NXMPNT:=0
137352
137352    MESSREAD: IF MXMPNT=0 THEN ENIMPLEMENT; EXIT FI
137356            IF 5D12=-1 THEN
137362               CMPNT
137363            ELSE
137364               A+1=:X:=CMPNT
137367               DO WHILE X><0
137370                  IF A=FMPNT THEN
137373                     A:=MXMPNT-FMPNT=:D:=0; T:=5MESSIZE; *RDIV ST
137401                     A-1*5MESSIZE+FMPNT
137404                  ELSE
137405                     A-5MESSIZE
137406                  FI; X-1
137407               OD
137410            FI; EXITA
137411    *)FILL
137413
137413    INTEGER ARRAY 5TSLSTATUS(0); * *+NU5PR/
137416    INTEGER ARRAY 5TSLNTIME(0); * *+NU5PR/
137421    * 5TSLS<*; )ZERO
137421    INTEGER ARRAY 5TSLCOUNTA(0); * *+NU5PR/
137424    * MXM1=-1; 5TSLC<*; )ZERO MXM1; )KILL MXM1
137424    INTEGER 5LOGPROC
137425    DOUBLE 5HISTART
137427    DOUBLE ARRAY 5HIDATA(100)
137627    DOUBLE 5HIOUTSIDE
137631
137631    INTEGER ARRAY ADRPROC(0)      % ADDRESSES OF PROCESS DESCRIPTIONS
137631    *       S500S
137632    *       X5=S500S+5SEMS
137632    * 8PR01
137632    *       X5
137633    *       X55=X5+5SEMS; )KILL X5; X5=X55; )KILL X55
137633    * 8PR02
137633    *       S500E
137634
137634    *)KILL YENDC; YENDC=*
137634    *7ENDC/
075077    INTEGER ARRAY N500REENT:=(C5REENT)
075100    INTEGER C5REENT
075101    INTEGER 5HRTP,5HRTB,5HINTERVAL,5HICHANNELS,5HIFLAG,CTTIFIELD
075107    INTEGER 5MLOG
075110    INTEGER 5MLOPROC                  % PROCESSNR OF PROCESS BEING MONITOR CALL LOGGED
075111                                      % 0 IF NONE          (CHECKED BY  MCHANDEL)
075111                                      % -1 IF ALL
075111    *)FILL
075111    *)KILL 7ENDC; 7ENDC=*
075111    *7ENDC/
137634
137634    INTEGER CMSADDR=?,5MSGI=?
137634
137634    %=========================================================
137634    %    CHHIMESS
137634    %
137634    %       ALLOCATE MESSAGE BUFFERS FOR HISTOGRAM MESSAGE AND WATCHDOG MESSAGE
```

```
137634
137634   CHHIMESS:
137634   % - ALLOCATE THE HISTOGRAM MESSAGE
137634         DO
137634         WHILE HIMESS=0                          % UNTIL HISTOGRAM MESSAGE ALLOCATED
137636               CMSADDR=:HIMESS                   % LET'S TRY TO PUT A HISTOGRAM MSG HERE
137640               A:=CMSADDR+"HIMSIZE-1"            % COMPUTE NEW FREE POINTER
137642               T:=CMSADDR; A=:CMSADDR
137644               IF A SHZ -12><T SHZ -12 THEN      % CROSSING PAGE BOUNDARY?
137650                     A SHZ 12=:CMSADDR           % SET FREE POINTER TO PAGE BOUNDARY
137652                     0=:HIMESS                   % FORGET THIS ONE, IT DID NOT FIT
137653               FI
137653               IF 5MSGI=0 THEN EXIT FI           % DO NOT BOTHER; NEW CHANCE LATER
137656         OD
137657   % - ALLOCATE WATCHDOG MESSAGES
137657         FOR X:="S5CPUDF" STEP 5CPUDFSTZ TO "E5CPUDF" DO      % ALL CPUS
137663            DO
137663            WHILE X.WATCHDOG=0                                % UNTIL WATCHDOG ALLOCATED
137665               A:=CMSADDR=:X.WATCHDOG
137667               T:=CMSADDR; A+"HIMSIZE-1"=:CMSADDR
137672               IF A SHZ -12><T SHZ -12 THEN                   % WILL IT CROSS PAGE BOUNDARY?
137676                     A SHZ 12=:CMSADDR                        % SET TO NEXT PAGE BOUNDARY
137700                     0=:X.WATCHDOG                            % MARK AS NOT YET ALLOCATED
137701               FI
137701               IF 5MSGI=0 THEN EXIT FI                        % WE WILL GET A NEW TRY LATER
137704            OD
137705         OD; EXIT
137710   *)FILL
137712
137712   INTEGER CMSADDR                % FIRST FREE LOCATION AFTER MESSAGE-BUFFERS
137713   INTEGER 5MSGI                  % USER MESSAGES INITIALIZED; WATCHDOG AND HISTOGRAM MAY REMAIN
137714
137714   %=============================================================
137714   %       MSINIT
137714   %
137714   %       INITIALIZE MIGRO-PROGRAM MESSAGE BUFFERS
137714   %       D=FIRST PHYSICAL PAGE FOR MESSAGES
137714
137714   INTEGER CCXX                   % SAVE FOR X
137715   INTEGER POINTER LEXIT
137716   MSINIT:
137716         T:=L=:"LEXIT"
137720         A:=0; AD SH 12; A=:5MBBANK:=D=:CMSADDR
137725   % - CLEAR POINTERS TO HISTOGRAM AND READ-MICRO-PGM-VERSION (WATCHDOG) MESSAGES
137725         0=:HIMESS
137726         FOR X:="S5CPUDF" STEP 5CPUDFSZ TO "E5CPUDF" DO 0=:X.WATCHDOG OD
137735   % - INDICATE MESSAGES FOR PROCESSES NOT INITIALIZED
137735         0=:5MSGI
137736   % - ALLOCATE MESSAGE BUFFER TO EACH PROCESS
137736         FOR X:="S500S" STEP 5SEMSIZE TO "S500E" DO           % PROCESS DESCRIPTIONS
137742            X=:CCXX
137743            IF CMSADDR+"5MESSIZE-1" SHZ -12><CMSADDR SHZ -12 THEN      % WILL NEXT MSG CROSS PAGE BOUNDARY?
137752                CALL CHHIMESS                                          % TRY TO PUT IN A HISTOGRAM MSG
137753            FI
137753            IF CMSADDR+"5MESSIZE-1" SHZ -12><CMSADDR SHZ -12 THEN      % WILL NEXT MSG CROSS PAGE BOUNDARY?
137762                A  SHZ 12 =: CMSADDR                                   % SET TO NEXT PAGE BOUNDARY
137764            FI
137764            CMSADDR=:CCXX.MESSBUFFR                                    % UPDATE PD
137767            A+5MESSIZE=:CMSADDR                                        % INCREMENT FREE POINTER
137771         OD
```

```
137773    % -
137773            "S500S".MESSBUFFR=:SWMSG
137776            1=:5MSGI                                % MESSAGE TO SWAPPER
140000            CALL CHHIMESS                           % TELL WE HAVE INITIALIZED PROCESSES (USED BY CHHIMESS)
140001            A:="S500E+5SEMSIZE-S500S"=:D:=0         % ALLOCATE REMAINING HISTOGRAM AND WATCHDOG MESSAGES
140004            T:=5SEMSIZE; *RDIV ST
140006            A=:CMXACTPROC                           % COMPUTE NUMBER OF PROCESSES
140007    % - CLEAR MESSAGE BUFFERS
140007            X:=SWMSG; A:="MSGMX"; D:=0; AD SHZ -12; IF D><0 THEN A+1 FI
140016            A SH 12+X-2=:L; T:=5MBBANK; A:=0; D:=0
140025            DO WHILE X<<=L
140027                *STDTX; AAX 2; JXZ ENDOD
140032            OD
140033    ENDOD:
140033    % -                                               % IF MSGS ENDS AT BANK BOUNDARY
140033            X:=HIMESS; CALL INMESS
140035            FOR X:="S5CPUDF" STEP 5CPUDFSZ TO "E5CPUDF" DO
140041                X=:CCXX:=X.WATCHDOG; T:=5MBBANK
140044                -1;      *STATX X5SND
140046                3RMICV; *STATX XMICF
140050                X:=CCXX                             % READ MICRO PROGRAM VERSION
140051            OD
140053            FOR X:="S500S" STEP 5SEMSIZE TO "S500E" DO
140057                X=:A:=X.MESSBUFFR; T:=5MBBANK; *AAX XADPR; STATX
140064                X:=A
140065            OD; GO LEXIT
140070    *)FILL
140102
140102    INMESS: T:=5MBBANK; A:=-1; *STATX X5SND         % SENDER
140105            3RPREG; *STATX XMICF                    % HISTOGRAM (READ P-REG)
140107            EXIT
140110    *)FILL
140111
140111    *)KILL YENDC; YENDC=*
140111    *7ENDC/
075111
075111    INTEGER HIMESS                          % ADDRESS OF HISTOGRAM MESSAGE
075112    INTEGER 5MSINIT                         % ND-500 SYSTEM INITIALIZATION FLAG
075113                                            % 0: ND-500 NOT INITIALIZED
075113                                            % 1: MESSAGES INITIALIZED
075113                                            % 2: COMPLETLY INITIALIZED
075113
075113    INTEGER SWMSG                           % MESSAGE TO SWAPPER ----------
075114
075114    INTEGER CMXACTPROC,TS5STOP
075116    INTEGER ARRAY RTPWORKA(65)
075203
075203    *)KILL ACCE5; ACCE5=0
075203    * 8PR09; XY5=ACCE5+16; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8PR19; XY5=ACCE5+16; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8PR29; XY5=ACCE5+16; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8PR39; XY5=ACCE5+16; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8PR49; XY5=ACCE5+16; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8PR59; XY5=ACCE5+16; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"-8PR10 8N500; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203      *"8CP52 -8PR20; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8CP53 -8PR30; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8CP54 -8PR40; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8CP55 -8PR50; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
075203     *"8CP56 -8PR60; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
```

```
"075203    *"8CP57; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
"075203    *"8CP58; XY5=ACCE5+HIMSI; )KILL ACCE5; ACCE5=XY5; )KILL XY5; "
"075203
075203    *"8N500; MSGMX=NN5MS+ACCE5; "          % SIZE OF ND-500 MESSAGE AREA
""075203
075203    RBUS
075203    *"8N500 8F5UD
"075203    INTEGER 5FXBNK                 % MEMORY BANK OF FIX-INFO TABLE AND DATA CAP. TABLE
075204    INTEGER 5FXTBL                 % ADDR OF FIX-INFO TABLE
075205    INTEGER 5DSPS                  % ADDR OF DATA CAPABILITY TABLE
075206
075206    *"8N500
"075206    INTEGER 5MBBANK                 % BANK NR OF MESSAGE BUFFERS
075207    INTEGER CERNENABLE             % SET IF MON CERN (376) MAY BE USED
075210    INTEGER 5DBFLAG
075211    *)KILL 7ENDC; 7ENDC=*
075211    *YENDC/
140111    *"
"140111    * 5TTIF=TTIFI
140111    * 5BCHF=BCHFL
140111    * 5PASS=PASST
140111    * 5SPAS=SPAST
140111    * 5CSTC=CSTCK
140111    @ELIB
140111
140111    *"-8N500
"140111
140111    @DEV 1
140111    @DEV (S-S-J)MRES-CDR1
```

```
140111   %
140111   %=======================================================================
140111   %       M R E S - C D R 1
140111   %=======================================================================
140111
140111
140111   *)MODIN
140111   %=========================================================================
140111   % 38.1        M O D I N    M O D U T    M O D I 1
140111   %
140111
140111   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
140111   %    SYNCRONOUS MODEM DRIVER                       %
140111   %    AND HANDLING ROUTINES FOR SINTRAN III         %
140111   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
140111
140111   SUBR MODIN,MODUT,MODI1
140111
140111   DISP -12
140111   INTEGER MOWSTAT
140111   INTEGER TERSW,MOSW
140111   PSID
140111
140111   SYMBOL EBSOH=1, EBSTX=2, EBETX=3, EBDLE=20, EBIUS=37, EBETB=46
140111   SYMBOL EBEOT=67, EBSYN=62, EBPAD=377
140111
140111   SYMBOL ASSOH=1, ASSTX=2, ASETX=3, ASEOT=4, ASACK=6, ASBEL=7
140111   SYMBOL ASDLE=20, ASDC1=21, ASNAK=25, ASSYN=26, ASETB=27, ASUS=37
140111
140111   % MODEM INPUT DRIVER
140111
140111   MODIN:
140111          CALL ID12
140112   MODI1: T:=HDEV+DDR;*EXR ST
140115          A=:LAST
140116          IF T:=CFREE=0 GO MOSWX
140121
140121   @ICR;
140121          T:=MOSW GOSW MOSW0, FAR IBM1, FAR IBM2, MOSW3, MOSW4, MOSW4,   % 0-5
140131                       FAR I3271, HASP1, HASP2, HASP3, FAR HASP4,         % 6-12
140136                       FAR HASP5, FAR HASP6, FAR HASP7, FAR HASP8,        % 13-16
140142                       FAR HASP9;                                         % 17
140143   @CR;
140143
140143   MOSW4: GO MODIN
140144
140144   MOTERMF:
140144          LAST                                % TERMINATOR FOUND
140145   MOSW3: T:=MBSYMOD; CALL TDBPUT
140147   MOSWX: CALL RTACT
140150          IF TERSW=7 OR =12 OR =14 GO MOSWY   % NTR/VIP-BTH/UTS-400
140162          T:=HDEV+DCONT; 2204; *EXR ST
140166          0=:TMR; 4=:MOSW                     % IGNORE NEXT CHARACTER
140171          GO MODIN
140172   MOSWY: T:=HDEV+DCONT; 2204; *EXR ST
140176          205; *EXR ST                        % READ NEXT CHARACTER (IF SYNC)
140200          0=:TMR=:MOSW
140202          GO MODIN
140203
140203   *)FILL
```

```
140223
140223    % TEST FOR TERMINATION CONDITION
140223
140223    @ICR;
140223    MOSW0: T:=TERSW GOSW FAR GEN0, FAR DCT0, FAR CDC0, FAR GER0, IBM0, VIP0, HASP0,
140234                        FAR NTR0, FAR MSV20, FAR I3270, FAR BTH0, FAR SSAAB,
140241                        FAR UTS0;
140242    @CR;
140242
140242    %       HASP WS :
140242    %
140242    %       MOSW=0, TERSW=6 : DECODE 1. CHARACTER (SHOULD BE SYN)
140242    %
140242    HASP0: IF A><EBSYN GO MOSWY                          % ERROR: RESYNC
140245           7=:MOSW; GO FAR MORET                         % READ SECOND
140250
140250    %
140250    %       MOSW=7 : DECODE 2. CHARACTER (SHOULD ALSO BE SYNC)
140250    %
140250    HASP1: IF A><EBSYN GO FAR MOSWY                      % ERROR : RESYNC
140253           10=:MOSW; GO FAR MORET
140256    %
140256    %       MOSW=10 : DECODE BLOCK FOLLOWING SYNC
140256    %
140256    HASP2: IF A=EBSYN GO FAR MORET
140261           IF A=EBPAD GO FAR MOSWX                       % END OF BLOCK
140264           T:=MBSYMOD; CALL TDBPUT
140266           IF A=EBDLE THEN                               % ACK OR TRANSP. TEXT
140271              11=:MOSW
140273           ELSE
140274              IF A=EBSOH THEN                            % ENQ OR NON-TRANSP. TEXT
140277                 12=:MOSW
140301              FI
140301           FI
140301           GO FAR MODPIN
140302
140302    %
140302    %       MOSW=11 : TRANSPARENT TEXT OR ACK (STARTS WITH DLE)
140302    %
140302    HASP3: T:=MBSYMOD; CALL TDBPUT
140304           IF A=EBSTX THEN                               % TRANSP. TEXT-BLOCK
140307              13=:MOSW
140311           ELSE
140312              GO MOSWX                                   % ACK
140313           FI
140313           GO FAR MODPIN
140314
140314    %
140314    %       MOSW=12 : NON-TRANSPARENT TEXT BLOCK OR ENQ (STARTS WITH SOH)
140314    %
140314    HASP4: T:=MBSYMOD; CALL TDBPUT
140316           IF A=EBSTX THEN                               % TEXT-BLOCK WITH CRC FOLLOWS
140321              15=:MOSW
140323           ELSE
140324              GO MOSWX                                   % ENQ
140325           FI
140325           GO FAR MODPIN
140326
140326    %
140326    %       MOSW=13 : TRANSPARENT TEXT BLOCK ENDED WITH DLE+ETB+CRC+CRC+PAD
```

```
140320    %
140326    HASP5:  T:=MBSYMOD; CALL TDBPUT
140330            IF A=EBDLE THEN MIN MOSW FI                    % ETB MAY FOLLOW
140334            GO FAR MODPIN
140335
140335    %
140335    %       MOSW=14 : DLE WAS READ IN TRANSP. TEXT MODE, CHECK IF NEXT IS ETB
140335    %
140335    HASP6:  T:=MBSYMOD; CALL TDBPUT
140337            IF A=EBETB THEN
140342                16=:MOSW                                   % ETB : 2 CRC FOLLOWS
140344            ELSE
140345                13=:MOSW
140347            FI                                             % RESET DLE-MODE
140347            GO FAR MODPIN
140350
140350    %
140350    %       MOSW=15 : NON-TRANSPARENT TEXT BLOCK ENDED WITH ETB+CRC+CRC+PAD
140350    %
140350    HASP7:  T:=MBSYMOD; CALL TDBPUT
140352            IF A=EBETB THEN MIN MOSW FI
140356            GO FAR MODPIN
140357
140357    %
140357    %       MOSW=16 : 1. CRC CHARACTER
140357    %
140357    HASP8:  T:=MBSYMOD; CALL TDBPUT
140361            MIN MOSW; GO FAR MODPIN                        % READ NEXT CRC
140363
140363    %
140363    %       MOSW=17 : 2. CRC CHARACTER
140363    %
140363    HASP9:  T:=MBSYMOD; CALL TDBPUT
140365            GO FAR MOSWX                                   % END OF BLOCK
140366    *)FILL
140407
140407    %       VIP CONTROL (HONEYWELL-BULL INTERACTIV TERMINALS)
140407
140407    VIP0:   IF A=0 OR =ASSYN GO FAR MORET
140413            T:=MBSYMOD; CALL TDBPUT; A/\177
140416            IF A=ASETX THEN
140421                T:=2=:MOSW; GO FAR MODPIN                  % BCB WILL FOLLOW AFTER ETX
140424            FI
140424            IF A=ASEOT GO FAR MOSWX                        % EOT : END OF BLOCK
140427            GO FAR MODPIN
140430
140430    % IBM 3780/2780
140430
140430    IBM0:   IF A=EBSYN GO FAR MORET
140433            T:=MBSYMOD; CALL TDBPUT
140435            IF A=EBETX OR =EBETB OR =EBIUS THEN            % ETX,ETB,IUS : 2 CRC WILL FOLLOW
140446               MIN MOSW                                    % MOSW=1
140447            FI
140447            IF A=EBPAD GO FAR MOSWX                        % PAD : END OF BLOCK
140452            GO FAR MODPIN
140453    %
140453    %       MOSW=1 : READ 1. CRC-CHARACTER
140453    %
140453    IBM1:   T:=MBSYMOD; CALL TDBPUT
140455            MIN MOSW; GO FAR MODPIN
```

```
140457   %
140457   %       MOSW=2 : READ 2. CRC-CHARACTER
140457   %
140457   IBM2:   T:=MBSYMOD; CALL TDBPUT
140461           0=:MOSW; GO FAR MODPIN                          % END OF BLOCK, READ PAD
140463
140463   %       GERTS-115
140463
140463   GER0:   IF A=0 OR =ASSYN OR =377 GO FAR MORET
140472           T:=MBSYMOD; CALL TDBPUT
140474           A/\177
140475           IF A=ASETX THEN T:=3=:MOSW FI                   % ETX : BCB WILL FOLLOW NEXT
140502   *'BKGS
"140502           GO FAR MODPIN
140503   %
140503   %       UNIVAC NTR
140503   %
140503   NTR0:   T:=MBSYMOD; CALL TDBPUT
140505           IF A=ASETX THEN 3=:MOSW FI                      % ETX : BCB WILL FOLLOW NEXT
140512           GO FAR MODPIN
140513
140513   *)FILL
140522
140522   %
140522   %       CD-200-USER
140522   %
140522   CDC0:   GO GER0
140523   %
140523   %       NOT DEFINED
140523   %
140523   GEN0:   GO GER0
140524   %
140524   %       DCT-2000
140524   %
140524   DCT0:   IF =0 OR =ASSYN OR =377 GO FAR MORET
140533           T:=MBSYMOD; CALL TDBPUT
140535           A/\177
140536           IF A=ASETX OR =ASEOT OR =ASACK OR=ASNAK OR=ASBEL OR =ASDC1 THEN
140560              3=:MOSW                                      % AFTER ETX,EOT,ACK,NAK,BELL,DC1 : 1 BCC
140562           FI
140562           GO FAR MODPIN
140563   %
140563   %       SIEMENS MSV2
140563   %
140563   MSV20:  IF A=ASSYN GO MORET
140566           T:=MBSYMOD; CALL TDBPUT; A/\177
140571           IF A=ASETX OR =ASETB OR =ASUS THEN MIN MOSW FI       % READ 2 CRC NEXT
140603           IF A=177 GO FAR MOSWX                                % END OF BLOCK
140606           GO MODPIN
140607
140607   %
140607   %       IBM-3270 EMULATOR
140607   %
140607
140607   I3270:  IF A=EBSYN GO MORET
140612           T:=MBSYMOD; CALL TDBPUT
140614           IF A=EBSTX THEN T:=6=:MOSW FI                   % STX : SET TEXT MODE
140621           IF A=EBEOT GO FAR MOSWY                         % EOT : NEW SYNC MAY FOLLOW
140624           IF A=EBPAD GO FAR MOSWX                         % ENDING PAD
140627           GO MODPIN
```

```
140630    %
140630    %       MOSW=6 : 3270 IN TEXT MODE
140630    %
140630
140630
140630    I3271:  T:=MBSYMOD; CALL TDBPUT
140632            IF A=EBETX OR =EBETB THEN T:=1=:MOSW FI    % ETX/ETB : 2 CRC WILL FOLLOW
140642            GO MODPIN
140643
140643    %
140643    %       BTH/VIP 7750 INTERACTIVE TERMINALS
140643    %
140643    BTH0:   IF A=0 GO MORET
140644            IF A=377 GO FAR MOSWX
140647            T:=MBSYMOD; CALL TDBPUT                    % PAD : END OF BLOCK
140651            A/\177
140652            IF A=ASETX OR =ASETB THEN                  % ETX/ETB : BCC NEXT
140660               2=:MOSW; GO MODPIN
140663            FI
140663            IF A=ASEOT GO FAR MOSWX                    % EOT
140666            GO MODPIN
140667
140667    *)FILL
140677
140677    %
140677    %       STANSAAB, ISO-1745 LINE PROCEDURE.
140677    %
140677
140677    SSAAB:  T:=177/\A=:D
140702            IF D=0 OR D=ASSYN GO MORET
140707            IF D=ASEOT OR D=ASNAK GO FAR MOSW3         % EOT/NAK : END OF BLOCK
140715            T:=MBSYMOD; CALL TDBPUT; A/\177
140720            IF A=ASDLE OR A=ASETX THEN 3=:MOSW FI      % DLE/ETX : 1 BCC WILL FOLLOW
140730            GO MODPIN
140731
140731    %
140731    %       UNIVAC UTS-400 EMULATOR
140731    %
140731    UTS0:   IF A=0 OR =ASSYN OR =377 GO FAR MORET      % UTS-400
140740            T:=MBSYMOD; CALL TDBPUT; A/\177
140743            IF A=ASETX THEN 3=:MOSW; FI                % AFTER ETX : 1 BCC
140750            GO MODPIN
140751
140751    MODPIN:
140751    MORET:  205; T:=HDEV+DCONT; *EXR ST
140755            -2=:TMR
140757            GO MODIN
140760    %
140760    % DRIVER FOR MODEM OUTPUT
140760    MODUT:  DO;*IOF
140761               IF BHOLD=0 THEN 0=:TMR=:MOWSTAT; T:=0; CALL RTACT; GO HIT FI
140770               T:=MBSYMOD; CALL TDBGET; T:=HDEV+DDW;*EXR ST
140775               TTMR=:TMR
140777               T:=5
141000    HIT:       T=:A; T:=HDEV+DCONT;*EXR ST;ION
141005               CALL ID10
141006            OD
141007
141007    RBOS
141022
```

```
141022
141022
141022
141022    *"8HM01+8HM02+8HM03+8HM04+8HM05+8HM06
"141022
141022    *"99NTB
"141022
141022    @DEV 1
141022    @DEV (S-S-J)MRES-CDR2
141022    %
141022    %=============================================================================
141022    %       M R E S - C D R 2
141022    %=============================================================================
141022
141022
141022    *"8MT1+8MT2+8MT3+8MT4
"141022    %=============================================================================
141022    % 39.1          C T R M A G T    M B U S Y    M F I N I    M F E I L
141022    %              C T R 2 M A G T    X F I N    X M B U S Y
141022    %
141022
141022    % LEVEL 11 ROUTINE TO PERFORM MAG.TAPE TRANSFER
141022
141022    % ACTIVATED BY MTRANS WITH B=DTATFIELD, X=ABST PAR.LIST
141022    % PAR0: FUNCTION CODE (SEE DRIVER)
141022    % PAR1: CORE ADDRESS
141022    % PAR2: UNIT NUMBER
141022    % PAR3: NUMBER OF WORDS IF READ/WRITE
141022
141022    SUBR CTRMAGT,CTR2MAGT,MBUSY,MFINI,MFEIL,MFIN,XFIN,XMBUSY,RTMAG,TAPBY,TRFIN,UNRINITI
141022
141022    DISP 0
141022        DOUBLE POINTER DP1=P1
141022    PSID
141022
141022    @DEC
141022    SYMBOL DS1600=1600,D6250=6250,DS800=800
141022    @OCT
141022
141022    % SUBROUTINE TO PERFORM READ/WRITE OPERATION
141022
141022    MTRWOPER: A:=L=:"MTLRG"
141024              IF CTRG BIT 0 THEN                                    % WRITE
141027                    CTRG/\177700+20=:T; AD:=CADRG; X:=1
141035                    CALL MAGTRANS                                 % READ STATUS
141036                    IF MWRING/\X=MWSTAT THEN A:=175; GO MTLRG FI  % WRITE PROTECT VIOL
141045                    IF MLOAD/\X><0 AND "TRNSF"="HMAGT" THEN       % NOT ON TMAGT/SMAGT
141054
141054    % LOAD POINT, WRITE ERASE GAP BEFORE OPERATION
141054
141054                       CTRG/\177700+14=:T; AD:=CADRG; X:=1
141062                       CALL MAGTRANS; IF X BIT 4 GO MTERET
141065                    FI
141065                    CMWCNT=:MACOU
141067              ELSE
141070                    IF A=4 OR A=64 THEN                 % READ BACKWARDS
141076                       IF X:=CXRG><0 THEN X-1 FI
141101                       AD:=MEMAD; D+X; A:=A+C; AD=:MEMAD=:CMADR
141106                    FI; -1=:MACOU
141110              FI; GO EFI1; *)FILL
```

```
41114   EFI1:     FOR MACOU DO                                  % NUMBER OF TIMES TO WRITE ERASE GAP IN WRITE OPERATION
41114               CTACNS=:TACOU                               % NUMBER OF RETRIES
41116               FOR TACOU DO
41116                  IF X:="TRNSF"="SMAGT" THEN
41122                     CXRG \/ DWONO =: X                        %%% 24 BIT WORD-COUNT
41125                  ELSE
41126                     X := CXRG                                 %%% 16 BIT WORD-COUNT
41127                  FI
41127                  CALL GOMAGTRANS
41130                  IF X NBIT 4 GO MTOKRET
41132
41132   % READ OR WRITE ERROR. BACKSPACE AND TRY AGAIN
41132   % ADVANSE ONE RECORD IF READ BACKWARDS
41132
41132                  IF CTRG/\77=4 OR A=64 THEN T:=16 ELSE T:=15 FI
41145                  CTRG/\177700; T+A; AD:=CADRG; X:=1
41152                  CALL MAGTRANS; IF X BIT 4 GO MTERET
41155               OD
41157
141157  % REPEATED ERROR. WRITE ERASE AND TRY AGAIN IF WRITE FUNCTION:
141157
141157               IF CTRG BIT 0 THEN
141162                  CTRG/\177700+14=:T; AD:=CADRG; X:=1
141170                  CALL MAGTRANS; IF X BIT 4 GO MTERET
141173               FI
141173            OD
141175            X:=CXRG; CALL GOMAGTRANS                  % LAST TRY
141177            IF X NBIT 4 GO MTOKRET; GO OVER; *)FILL
141207    OVER:   IF "TRNSF"="TMAGT" AND CTRG NBIT 0 AND X BIT 16 THEN
141220                  CTACNS=:TACOU
141222                  FOR TACOU DO
141222                     CTRG/\177700+15=:T; AD:=CADRG; X:=1
141230                     CALL MAGTRANS; IF X BIT 4 GO MTERET
141233                     CTRG/\177700=:T
141236                     IF CTRG/\37=0 THEN T+23 ELSE T+24 FI
141244                     AD:=CADRG; X:=CXRG
141246                     CALL MAGTRANS; IF X NBIT 4 GO MTOKRET
141251                  OD; GO MTOKRET
141254            FI; GO MTERET
141255  MTERET: A:=0; GO MTLRG
141257  MTOKRET: MIN "MTLRG"; GO MTLRG
141261  *)FILL
141265
141265
141265  CTRMAGT:
141265  *"8MT1+8MT2+8MT3+8MT4 8DTMT
"141265    CTR2MAGT: X:=:B
141266            CALL PICKFPAR; AD=:X.MEMAD=:X.CMADR       % 24-BIT MEMORY ADRESS
141271            A SH 14+T=:X.CTRG                         % FUNCTION CODE
141274            CALL PICKLPAR; T=:X.CXRG                  % WORD-COUNT
141276            A=:X.CDRG; A:=D=:X."MRETURN"              % ADRESS OF 'WORDS READ'
141301            X.CTRG/\700 SHZ -6 \/ X.CDRG=:X.CDRG      % UNIT IN CDRG AND/OR
141306            X.CDRG SH 6\/X.CTRG=:X.CTRG               % BITS 8-6 IN CTRG
141312            IF X."TRNSF" = "SMAGT" THEN               % STC-DRIVES?
141316               0 =: X.DWONO                           % ZERO MOST SIGN. PART OF WORD-COUNT
141317               B=:D:=X
141321               A:=XUNIT (CDRG) =: XNOWUNIT            % GET INBT/OUTBT DATAFIELD
141324               X:=B;B:=D
141326            FI
141326            IF X.CTRG/\77=6 OR A=50 OR A=51 OR A>=60 AND A<=64 THEN % 24-BIT WORD-COUNT?
```

```
141347              IF X."TRNSF"><"SMAGT" THEN A:=201; B:=X; GO FAR MBF5ERR FI  % ONLY FOR STC.
141356              CALL PICKXLPAR                      % SPECIAL PARAM. FETCH!!!
141357              A=:X.NMTRECS                        % NUMBER OF RECORDS IN FUNCTION 50 & 51
141360              IF X.CTRG/\77 =50 OR A=51 THEN
141370                  0=: X.DWONO
141371              ELSE
141372                  X.NMTRECS=: X.DWONO             % MOST SIGN. WORD-COUNT
141374              FI
141374              A := D =: X.CXRG                    % LOWER HALF TO CXREG
141376          FI
141376          X=:B; 0=:CERRCODE=:HSTAT                % INITIATE
141401
141401          IF X:=CDRG>MAXUNIT THEN                 % X=MT UNIT NUMBER
141405              A:=173; GO FAR MBF5ERR; *)FILL      % ILLEGAL UNIT
141417          FI
141417
141417          IF CTRG/\77=23 THEN                     % SELECT PARITY AND DENSITY
141424              IF "TRNSF" = "SMAGT" THEN
141430                  IF CXRG = DS1600 THEN "0"       % TRANSFORM '1600' OR '6250' OR '800'
141435                  ELSE IF = D6250 THEN 1
141442                  ELSE IF = DS800 THEN 2 FI FI FI
141447                  A =: CXRG                       % STORE
141450              FI
141450              IF CXRG<0 OR A>6 THEN
141455                  A:=174; GO FAR FINER            % ILLEGAL PAR.
141457              FI; CXRG=:ADNSTY(X); GO FAR FIN     % DENSITY SELECTION
141462          FI
141462
141462          IF A=25 THEN                            % READ TAPE STATUS
141465              IF CXRG<4 THEN
141471                  -1=:HSTAT                       % BUFFER TOO SMALL!!
141473                  0=:MRETURN                      % 0=:MRETURN
141474                  GO FAR FIN
141475              FI
141475              AD:=MEMAD
141476              T:=RHSTAT(X); 0=:RHSTAT(X); CALL PSTWORD
141501              T:=RERRCOUNT(X); 0=:RERRCOUNT(X); CALL PSTWORD
141504              T:=WHSTAT(X); 0=:WHSTAT(X); CALL PSTWORD
141507              T:=WERRCOUNT(X); 0=:WERRCOUNT(X); CALL PSTWORD
141512              A:=4=:MRETURN                       % 4=:MRETURN
141514              GO FAR FIN; *)FILL
141525          FI
141525
141525          IF A=24 THEN                % READ LAST STATUS
141530              SHSTAT(X)=:HSTAT; GO FAR FIN
141533          FI
141533
141533          IF "TRNSF" = "SMAGT" AND CTRG/\77=36 THEN
141544              AD:=MEMAD
141545              FOR X := 0 TO 14 DO
141551                  T := FCST (X); CALL PSTWORD
141553              OD
141555              A:=15=:MRETURN; GO FAR FIN
141560          FI
141560
141560          IF A = 42 THEN                          % READ FORMAT/DENSITY CODE.
141563              ADNSTY(X)=:MRETURN; GO FAR FIN
141566          FI
141566
141566          IF A=6 THEN                             % SET RETRY COUNTER
```

```
141571                NMTRECS=:CTACNS; CXRG=:CMWCNT
141575                GO FAR FIN
141576             FI
141576
141576             ADNSTY(X)=:CDRG
141600
141600             IF TMR><0 THEN CALL ID11; FI                % WAIT FOR INTERRUPT OR TIME-OUT.
141603
141603             IF CTRG/\77=20 THEN                         % READ STATUS
141610                CTADRG; X:=1; CALL MAGTRANS; GO FAR FIN; *)FILL
141622             FI
141622             IF A>=10 AND A<26 GO FAR NOTRW
141630             IF A >= 33 AND A < 40 GO FAR NOTRW          % STC-CONTROLLER.
141636
141636    % READ OR WRITE
141636
141636          IF A=50 THEN                                  % READ MULTIPLE RECORDS
141641             IF CMWCNT=0 THEN MWCNT=:CMWCNT FI
141645             IF CTACNS=0 THEN TACNS=:CTACNS FI
141651             0=:MRECCOUNTER
141652             DO WHILE MRECCOUNTER><NMTRECS
141656                IF CMWCNT=0 THEN -1=:CMWCNT FI
141662                IF CTACNS=0 THEN -1=:CTACNS FI
141666                CALL FAR MTRWOPER
141667                GO F50ERR
141670                CMADR; A:=:D-MEMA2; *RDCR ADC DD
141674                A:=:D-MEMA1
141676                IF A><0 GO XER50                          % TOO LONG RECORD
141677                IF X:=MRECCOUNTER=0 THEN
141702                   A:=D=:CMTRECSIZE
141704                ELSE
141705                   IF CMTRECSIZE><D GO XER50              % DIFFERENT RECORD SIZE
141710                FI; MIN MRECCOUNTER; 0/\0
141712                CMADR=: MEMAD
141714             OD; GO FIN50
141716          FI
141716
141716          IF A=51 THEN                                  % WRITE MULTIPLE RECORDS
141721             IF CMWCNT=0 THEN MWCNT=:CMWCNT FI
141725             IF CTACNS=0 THEN TACNS=:CTACNS FI
141731             0=:MRECCOUNTER
141732             DO WHILE MRECCOUNTER><NMTRECS
141736                IF CMWCNT=0 THEN -1=:CMWCNT FI
141742                IF CTACNS=0 THEN -1=:CTACNS FI
141746                CALL FAR MTRWOPER
141747                GO F51ERR
141750                MIN MRECCOUNTER; 0/\0
141752                CMADR=: MEMAD
141754             OD; GO FIN
141756          FI
141756
141756    % OTHER READ/WRITE FUNCTIONS
141756
141756          TACNS=:CTACNS; MWCNT=:CMWCNT
141762          CALL FAR MTRWOPER; GO EEFIN; GO FINX
141765    *)FILL
141767
141767    XER50:   CTRG SHZ -6/\7=:X; -1=:HSTAT=:SHSTAT(X)  % HSTAT=-1 MEANS READING RECORDS OF DIFFERNET SIZES
141776    F50ERR:
141776    FIN50:   A:=CMTRECSIZE=:D:=MRECCOUNTER
```

```
142001            AD=:DMRETURN; GO FIN                          % NUMBER OF RECORDS AND RECORDSIZE
142003    F51ERR: A=:D
142004            MRECCOUNTER=:MRETURN                          % UPDATE NUMBER OF RECORDS WRITTEN WITHOUT ERRORS
142006            A:=D
142007    EEFIN:  IF A=0 GO FIN
142010           GO FINER
142011
142011    FINX:     A=:X:=CTRG/\77:=:X
142015            IF X<60 OR X>63 THEN                        % IF NOT DOUBLE WORD-COUNT
142023              IF X=4 THEN
142026                IF MEMA2-CMAD2>>CXRG THEN CXRG FI
142034              ELSE
142035                IF A-MEMA2>>CXRG THEN A:=T FI
142042              FI; IF X=26 THEN A SH 1 FI       % NUMBER OF BYTES
142046              A=:MRETURN
142047            ELSE
142050              IF X=64 THEN
142053                 MEMAD; A:=:D-CMAD2; *RDCR ADC DD
142057                 A:=:D-CMAD1
142061              ELSE
142062                 CMADR; A:=:D-MEMA2; *RDCR ADC DD
142066                 A:=:D-MEMA1
142070              FI; IF X=62 THEN AD SH 1 FI                 % NUMBER OF BYTES
142074              AD=:DMRETURN
142075            FI; GO FIN; *)FILL
142100     MFIN:
142100     FIN:
142100    *"8MT1+8MT2+8MT3+8MT4 8DTMT
"142100
142100              0=:TMR; IF RTRES><0 THEN CALL RTACT; FI
142104    *"8MT1+8MT2+8MT3+8MT4
"142104              CALL ID11; *JMP *-1
142106    MBF5ERR: A=:CERRCODE
142107           IF CTRG/\77=50 OR A=51 THEN 0=:MRETURN FI
142120           GO XFIN
142121    FINER:  A=:CERRCODE
142122    XFIN:   HSTAT BONE 4=:HSTAT     % SET OR-OF-ERRORS
142125           GO FIN
142126
142126
142126    *)FILL
142132
142132    % ANOTHER FUNCTION THAN READ OR WRITE
142132
142132     NOTRW: IF A=12 THEN
142135
142135    % WRITE EOF. ERASE FIRST IF LOAD POINT
142135
142135                CTRG/\177700+20=:T; AD:=CADRG; X:=1
142143                CALL MAGTRANS
142144                IF MWRING/\X=MWSTAT THEN A:=175; GO FINER; FI
142153                IF MLOAD/\X><0 AND "TRNSF"="HMAGT" THEN
142162                   CTRG/\177700+14=:T; AD:=CADRG; X:=1
142170                   CALL MAGTRANS; IF X BIT 4 GO FIN
142173              FI FI
142173    % PERFORM OPERATION:
142173
142173              CTADRG; X:=1; CALL MAGTRANS; IF X BIT 4 GO FIN
142200
142200    % BACSPACE EOF-MARK IF TANDBERG TAPE AND REVERSE TO EOF OPERATION:
```

```
42207
42210             IF "TRNSF"="TMAGT" AND CTRG/\77=11 THEN
42211                CTRG/\177700+15=:T; AD:=CADRG; X:=1
42217                CALL MAGTRANS
42220             FI
42230             GO FIN
42221    *)FILL
42226    % TRANSFER SUBROUTINE:
42226
42226     GOMAGTRANS: IF CTRG/\77>=50 AND <=64 THEN
42236                    IF A=50 OR A=51 THEN
42244                          A-50
42245                    ELSE IF A=60 OR A=61 OR A=64 THEN
42257                          A-60
42260                    ELSE IF A=62 OR A=63 THEN
42267                          A-34
42270                    FI FI FI
42270                    A=:T; CTRG/\177700; T\/A
42274                    AD:=CADRG
42275                 ELSE CTADRG
42277                 FI
42277     MAGTRANS: TAD=:TADRG; X=:XRG
42301               X:=L=:"CLRG"; X:=XRG
142304              DO
142304                    TTMR=:TMR
142306                    CALL TRNSF; GO ERROR; GO BUSY; GO FINISH
142312     MBUSY:         TAD=:TADRG; X=:XRG
142314                    TRG/\700 SHZ -6 +"SHSTAT"+B=:X; HSTAT=:X.SO  % SAVE LAST STATUS
142324                    IF "TRNSF" = "SMAGT" THEN
142330                       CALL MFRESAVE
142331                    FI
142331    *"8MT1+8MT2+8MT3+8MT4 8DTMT
142331                    CALL ID11
142332    *"8MT1+8MT2+8MT3+8MT4
142332      XMBUSY:       TADRG; X:=XRG
142334              OD
142335     MFINI: 0=:TMR; A=:D
142337              X BZERO 4=:HSTAT
142341              TRG/\700 SHZ -6=:X; HSTAT=:SHSTAT(X)=:X          % SAVE STAT.
142350              A:=D; GO CLRG; *)FILL
142361
142361    % ERROR IN MAG.TAPE OPERATION
142361
142361    INTEGER AAREG =?
142361    MFEIL:  0=:TMR; X BONE 4=:HSTAT; A=:AAREG
142365              TRG/\700 SHZ -6=:X; HSTAT=:SHSTAT(X)=:X          % SAVE STAT.
142374              MIN ERCNT; 0/\0; T:=AERRB\/X=:AERRB
142401              IF CTRG/\77=0 OR A=2 OR A = 26 OR A=50 OR A=60 OR A=62 THEN
142423                    IF HSTAT/\BADTAPE><0 THEN
142426                       CTRG SHZ -6/\7=:X
142432                       HSTAT\/RHSTAT(X)=:RHSTAT(X); MIN RERRCOUNT(X); 0/\0
142437                    FI
142437              ELSE IF A=1 OR A=3 OR A = 27 OR A=51 OR A = 61 OR A = 63 THEN
142462                    IF HSTAT/\BADTAPE><0 THEN
142465                       CTRG SHZ -6/\7=:X
142471                       HSTAT\/WHSTAT(X)=:WHSTAT(X); MIN WERRCOUNT(X); 0/\0
142476                    FI
142476                 FI
142476              FI
142476              GO CONT1
```

```
142477    *IFILL
142502    INTEGER AAREG
142503
142503    CONT1:    X:=HSTAT
142504              IF TRG/\77<10 OR A>22 THEN
142514                    IF SERRB/\X><0 THEN
142517                      IF HSTAT BIT 11 AND A/\BADTAPE><0 GO RETR    % EOT&BAD DATABLOCK
142524                      IF X BIT 13 THEN                             % DMA-ERROR
142526                        IF X BIT 7 OR X BIT 11 GO RETR             % DMA-ERROR & EOT/EOF
142532                      FI
142532                      GO CFINX
142533                    FI
142533                    IF BADTAPE/\X=0 GO CFINX                       % WARNING!!
142536                    GO RETR                                        % NOT SERIOUS ERROR
142537              ELSE
142540                    IF SERRB/\X><0 OR TRG/\77=22 THEN GO CFINX FI
142551              FI
142551    RETR:   X:=HSTAT
142552              IF CTRG/\77>=60 AND A<=63 THEN
142562                 0/\0; 0/\0                                        % PATCH FOR GECO
142564              FI; AAREG; GO CLRG
142566     CFINX:IF CTRG/\77 = 50 OR A=51 THEN          % ERROR IN MULT.RECORDS
142576                 GO MTLRG                          % RETURN TO CALLING LOOP
142577              ELSE
142600                    AAREG; GO FAR FINX            % ERRORS IN OTHER FUNCTIONS
142602              FI
142602     RBUS
142604
142604    *'99SM1+99SM2+99SM3+99SM4
"142604
142604    %%%%%  SUBROUTINE TO RESTART USER WITH ASYNC. TIMEOUT IF REWIND-BUSY!!!
142604
142604    SUBR MFRESAVE
142604
142604    INTEGER XSAVE, LSAVE
142606
142606    MFRESAVE:
142606              IF RTRES = 0 THEN EXIT FI                 % NOT RT-USER!!!
142611              X =: XSAVE := XNOWUNIT
142613              A := L =: LSAVE
142615              IF XRG >< 0 THEN                          % REWINDING!!
142617                 * IOF                                  % PROTECT!!!
142620                 0 =: TMR                               % DON'T USE DMA-TIMEOUT!
142621                 X.TTMR =: X.TMR
142623                 "SMTBREL"=:MFUNC; CALL XRTACT
142626                 * ION                                  % UNPROTECT!!!
142627              FI
142627
142627              X := XSAVE; A := LSAVE =: P               % RETURN TO CALEE
142632
142632    RBUS
142634
142634    *'8MT1+8MT2+8MT3+8MT4 8DTMT
"142634
```

```
142634
142634
142634     * CAMA
142634      %=============================================================================
142634     % 39.6        T F D I S K   F D I B U S   F D I F I N   F D I F E I L
142634     %             B U F D I S C   F I N F D I S C
142634     %
142634     %
142634     % ROUTINE TO SET UP CALL TO THE FLOPPY DISC DRIVER
142634     % LEVEL 11
142634     %
142634     % ENTRY: B=ADDRESS OF THE DATAFIELD
142634     %        X=ADDRESS OF THE PARAMETER LIST
142634     %
142634     %
142634     % PARAMETERS:
142634     %
142634     %      PAR-0(BIT 0-5)= FUNCTION CODE
142634     %
142634     %              0 - READ
142634     %              1 - WRITE
142634     %              2 - TEST PARITY (READ CRC)
142634     %              3 - DUMMY
142634     %              7 - ERASE TAPE
142634     %             10 - ADVANCE TO EOF
142634     %             11 - REVERCE TO EOF
142634     %             12 - WRITE EOF
142634     %             13 - REWIND TO BOT
142634     %             14 - ADVANCE ONE RECORD
142634     %             15 - REVERCE ONE RECORD
142634     %             21 - CLEAR-DEVICE
142634     %             20 - READ STATUS
142634     %             24 - READ LAST STATUS
142634     %             36 - READ EXTENDED STATUS
142634     %             41 - FORMAT FLOPPY
142634     %             42 - READ FORMAT
142634     %             43 - READ DELETED RECORD
142634     %             44 - WRITE DELETED RECORD
142634     %             46 - READ-CURRENT-ADDRESS
142634     %             47 - WRITE-NEW-ADDRESS
142634     %             54 - COPY-FLOPPY
142634     %             55 - FORMAT-TRACK
142634     %             56 - CHECK-CARTRIDGE (READ AND TEST CRC)
142634     %             57 - TEST CARTRIDGE CAPASITY (WRITE)
142634     %             70 - RETENSION CARTRIDGE
142634     %             71 - PERFORM TEST
142634     %             72 - ILLEGAL COMMAND
142634     %             73 - CLEAR
142634     %             74 - CONTINIOUS READ
142634     %             75 - CONTINIOUS WRITE
142634     %
142634     %       BIT 6-7: UNIT NUMBER  (0,1,2,3)
142634     %
142634     %       BIT 12: STREAMER CASSET
142634     %
142634     %       PAR-1 (24-BIT MEMORY ADDRESS)
142634     %
142634     %       PAR-2 (DISK ADDRESS (LOGIC SECTOR ADDRESS))
142634     %
142634     %       PAR-3 AMOUNT TO TRANSFER (LOGICAL BLOCKS)) IF READ/WRITE
```

```
142634   %              ELSE PAR-3=FORMAT NO. IF FUNCTION IS SELECT FORMAT
142634   %
142634   %  THE FUNCTION FLAGS ARE:
142634   %
142634   %       BITS 0 TO 10    = TIMEOUT IN SECONDS
142634   %                              777=MAX (SFTIM(100))
142634
142634   % SYMBOL   3FLOP = 11 % LEGAL ON FLOPPY
142634   % SYMBOL   3STRE = 12 % LEGAL ON STREAMER
142634   % SYMBOL   3FLTI = 13 % SHORT FLOPPY TIMEOUT (SFTIM (101)).
142634   % SYMBOL   3FRES = 14 % LEGAL ONLY FROM RT-PROGS ON RING 2
142634   % SYMBOL   3DOUA = 15 % DOUBLE ADDRESS; BITS 0-5 GIVES NEW FUNCTION TO USE
142634   % SYMBOL   3DOUB = 16 % DOUBLE AMOUNT; BITS 0-5 GIVES NEW FUNCTION TO USE
142634   % SYMBOL   3ILLF = 17 % ILLEGAL FUNCTION
142634
142634   @ICR
142634
142634   INTEGER ARRAY SFTIM := (
142634    7170,   7170,  23056,   1001,     -1,     -1,     -1,   2777,
142644    3700,   1700,   3020,   7200,     -1,   1001,   1001,     -1,
142654    3001,  33073,     -1,     -1,   3001,     -1,     -1,     -1,
142664      -1,     -1,     -1,     -1,     -1,     -1,   3010,     -1,
142674    1010,   1170,   1010,   1010,   1010,     -1,   1001,   1001,
142704      -1,     -1,     -1,     -1,  11400,   1100,   3777,   2777,
142714   21000,  21001,  23003,  21003,  42074,  42075,     -1,     -1,
142724    2777,  12777,   2001,   2777,  12200,  12200,     -1,     -1,
142734    1400,        % MAX TIMEOUT
142735      04);       % SHORT FLOPPY TIMEOUT
142736
142736   @CR;
142736
142736   INTEGER SFFUN                  % FLAGS AND TIMOUT COUNTER
142737
142737   SUBR TFDISK,FDIBUS,FDIFIN,FDIFEIL,BUFDISC,FINFDISC,FLV1STFL
142737
142737   % LOCAL SUBROUTINE TO CHANGE FUNCTION CODE WHEN THIS IS REQUIRED
142737   % ENTRY WITH D=DATAFIELD; EXIT WITH X=DATAFIELD
142737   CHFFUNC: X:=77; X/\T; X:=:D; X.CTRG/\177700\/D=:X.CTRG      % NEW FUNCTION
142746            X:=:D; SFTIM(X)=:SFFUN; X:=D
142752            EXIT
142753
142753   TFDISK: X:=:B
142754           0=:X.HSTAT=:X.CERRCODE
142756           CALL PICKFPAR; AD=:X.MEMAD; T=:X.CTRG
142761           X=:D;=77; X/\T; T:=SFTIM(X)=:SFFUN          % FLAGS ACCORDING TO FUNCTION
142766           IF T<0 GO FAR D201ER                        % ILLEGAL FUNCTION
142770           IF T BIT 3FRESTRICTED THEN                  % FUNCTION ONLY ALLOWED FROM RT-PROGS ON RING 2-3
142772              IF D.RTRES=0 GO FAR D201ER               % FLOPPY NOT RESERVED
142776              IF A.ACTPRI BIT 5BACKGR OR A/\3<2 GO FAR D201ER  % ONLY LEGAL FOR RT-PROGS ON RING 2
143006           FI
143006           IF T BIT 3DOUA THEN                         % DOUBLE FLOPPY DISC ADDRESS
143010              CALL CHFFUNC                             % USE ANOTHER FUNCTION
143011              CALL PICKYLPAR
143012              IF A><0 THEN X=:B; GO FAR 100ER FI       % OUTSIDE DEVICE LIMITS
143015              A:=D=:X.CDRG                             % FLOPPY DISC ADDRESS
143017           ELSE IF T BIT 3DOUB THEN                    % DOUBLE WORDCOUNT
143022              CALL CHFFUNC                             % USE ANOTHER FUNCTION
143023              CALL PICKXLPAR
143024              %% A=:X.DWONO                            % NOT USED NOW
143024              A:=D=:X.CXRG                             % WORDCOUNT
```

```
143026          ELSE
143027             X:=D; CALL PICKLPAR
143031             A=:X.CDRG:=D=:X."FRETURN"
143034          FI; FI; T=:X.CXRG
143035          B:=X; GO L1; *)FILL                          % B=ADDR OF DATAFIELD
143051
143051  L1:     IF CTRG/\700 SHZ -6>MAXUNIT GO FAR 173ER     % ILLEGAL UNIT NUMBER
143057          A=:CFLUN
143060
143060          IF CTRG NBIT 14 AND SFFUN BIT 3FLTI THEN     % NOT STREAMER AND SHORT TIMOUT
143066             SFTIM(101)
143070          ELSE
143071             IF SFFUN/\777=777 THEN SFTIM(100) FI
143100          FI; A-=:TTMR                                 % TIMEOUT IN SECONDS
143102          IF "TRNSF"="BFDIS" THEN -1 ELSE -3 FI
143111          A=:TACNS                                     % RETRY COUNTER
143112          X:=CFLUN                                     % X=FLOPPY UNIT NUMBER
143113          IF CTRG/\77 =24 OR A=20 THEN
143123             SHSTAT(X)=:HSTAT; GO FAR FIN              % READ STATUS/LAST STATUS
143126          FI
143126
143126          IF CTRG BIT 14 GO FAR RWOPER                 % STREAMER TAPE
143131
143131          IF A/\77<2 OR A=43 OR A=44 GO FAR RWOPER     % READ & WRITE
143143
143143          IF A=13 OR A=15 OR A=16 OR A=47 THEN         % POSITION THE DISC
143157             IF CTRG/\77=13 THEN A:=0 ELSE             % REWIND
143166                IF A=15 THEN NFDIADR(X)-1 ELSE         % BACKSPACE ONE RECORD
143174                   IF A=47 THEN CXRG ELSE              % SET CURRENT DISC ADDR
143201                      NFDIADR(X)+1                     % ADVANCE ONE RECORD
143203                   FI
143203                FI
143203             FI; T:=A; X:=FDIFORM(X)                   % T=NEW FLOPPY DISC ADDR; X=FLOPPY FORMAT NUMBER
143205             X:=LFADDR(X)                              % X=LAST ADDRESS FOR THIS FORMAT
143206             IF CTRG/\77-16=0 THEN X+1 FI              % ADVANCE ONE RECORD
143213             IF T>>=X GO 100ER                         % OUTSIDE DEVICE LIMITS
143215             T=:NFDIADR(CFLUN); GO FIN; *)FILL         % SET NEXT DISC ADDRESS
143231          FI
143231          IF A=40 THEN                                 % SELECT FORMAT
143234             IF X:=CXRG>>17 OR LFADDR(X)<0 THEN 174; GO FINER FI     % ILLEGAL FORMAT
143244             CXRG=:FDIFORM(CFLUN); GO FIN
143250          FI
143250          IF A=41 THEN                                 % FORMAT FLOPPY
143253             FDIFORM(X) SH 10+CTRG=:CTRG
143257             1=:CXRG; 0=:CDRG; TAD:=CTADRG; X:=CXRG; CALL CFDISK
143265             IF HSTAT BIT 4 THEN 223; GO FINER FI
143272             GO FIN
143273          FI; GO OVER; *)FILL
143276
143276  FINFDISC:
143276  FIN:    0=:TMR; IF RTRES><0 THEN CALL RTACT FI
143302          CALL ID11; GO ERR22
143304
143304  173ER:  A:=173; GO FINER
143306  D201ER: D=:B; A:=201; GO FINER
143311  D100ER: D=:B
143312  100ER:  A:=100                                       % OUTSIDE DEVICE LIMITS
143313  FINER:  A=:CERRCODE
143314  EFINER: HSTAT BONE 4=:HSTAT; GO FIN
143320
```

```
143320   OVER:  IF A=42 THEN                                  % READ FORMAT
143323               FDIFORM(X) SH 10+CTRG=:CTRG
143327               NFDIADR(X)=:CDRG; 1=:CXRG
143333               TAD:=CTADRG; X:=CXRG; CALL CFDISK
143336               IF HSTAT BIT 4 GO EFINER
143341               IF "TRNSF"="BFDIS" THEN A:=D ELSE A:=DRG FI
143350               A/\17=:FRETURN=:FDIFORM(CFLUN); GO FIN
143355          FI
143355          IF A=12 THEN                                 % WRITE EOF
143360               FDIFORM(X) SH 10+CTRG/\177700+5=:CTRG % FUNC=5 =WRITE DELETED RECORD
143366               1=:CXRG; A:=B=:CARG
143372               NFDIADR(X)=:CDRG+1=:NFDIADR(X)
143376               X:=FDIFORM(X); T:=LFADDR(X)             % T=LAST ADDRESS FOR THIS FORMAT
143400               IF CDRG>>T GO 100ER                     % OUTSIDE DEVICE LIMIT
143403               TAD:=CTADRG; X:=CXRG; CALL CFDISK; GO FIN
143407          FI
143407          IF A=46 THEN                                 % READ CURRENT DISC ADDR.
143412               NFDIADR(X)=:FRETURN; GO FIN
143415          FI
143415          IF A=54 THEN                                 % COPY FLOPPY DISKETTE
143420               CXRG/\3 SHZ 14                          % DESTINATION UNIT
143423               A\/CTRG=:CTRG                           % TO BITS 14-15 IN FUNC.
143425               GO FAR RWOPER
143426          FI
143426          IF A=10 THEN A:=1 ELSE                       % ADVANCE TO EOF
143433             IF A=11 THEN A:=-1 ELSE                   % REVERSE TO EOF
143440               IF A=3 GO FIN                           % COMPARE
143443               GO RWOPER; *)FILL                       % ILLEGAL FUNCTION??
143456             FI
143456          FI; A=:FDIMOD
143457          FDIFORM(X) SH 10+CTRG/\177700+2=:CTRG
143465          1=:CXRG; NFDIADR(X)=:CDRG
143471          IF FDIMOD<0 THEN CDRG-1=:CDRG FI
143476          DO                                           % UNTIL EOF IS FOUND OR UNTIL ERROR
143476               X:=FDIFORM(CFLUN); T:=LFADDR(X)         % T=LAST ADDRESS FOR THIS FORMAT
143501               IF CDRG>>T GO FAR 100ER                 % OUTSIDE DEVICE LIMITS
143504               X:=CXRG; TAD:=CTADRG; CALL CFDISK
143507               IF HSTAT BIT 4 GO FAR FIN
143512               IF A BIT 5 THEN                         % DELETED RECORD
143514                    IF FDIMOD>0 THEN A+CDRG=:CDRG FI
143521                    GO OUT
143522               FI; CDRG+FDIMOD=:CDRG
143525          OD
143526   OUT:   CDRG=:NFDIADR(CFLUN)
143531          HSTAT BZERO 4=:HSTAT; GO FAR FIN
143535   *)FILL
143541
143541   % READ AND WRITE FUNCTIONS
143541
143541   RWOPER:IF CTRG NBIT 14 THEN                          % IF NOT STREAMER
143544             FDIFORM(X) SH 10+CTRG=:CTRG
143550             X:=FDIFORM(X); T:=LFADDR(X)               % T=LAST ADDRESS FOR THIS FORMAT
143552             IF CDRG+CXRG>>T GO FAR 100ER              % OUTSIDE DEVICE LIMIT
143556          FI; TACNS=:TACOUNT
143560          FOR TACOUNT DO
143560             AD:=MEMAD=:CMADR; X:=CXRG; TAD:=CTADRG; CALL CFDISK
143565             IF CTRG/\77=43 THEN                    % DELETED RECORD
143572                IF HSTAT NBIT 5 THEN 231; GO FAR FINER FI  % ERROR, NOT DELETED RECORD
143577             ELSE
143600                IF HSTAT BIT 5 THEN 3; GO FAR FINER FI     % ERROR: END-OF-FILE
```

```
143615              FI; IF HSTAT NBIT 4 GO FAR FIN
143620           OD; GO FAR FIN
143633
143633      FDISK: TAD=:TADRG; X=:XRG; A:=L=:"CFLRG"
143637      FLV1STFL: TTMR=:TMR              % RESTARTED HERE FROM RT-PROG WHEN I/O FLOPPY (OLD FLOPPY)
143621           DO
143621                 X:=XRG; TAD:=TADRG
143623                 CALL TRNSF; GO ERROR; GO BUSY; GO FINISH
143627      FDIBUS:    CALL ID11                                    % RETURN HERE WHEN BUSY
143630           OD
143631
143631      % SAVED FOR EACH ERROR RETURN FROM DRIVER
143631      INTEGER LEHSTAT                          % HSTAT WHEN LAST ERROR RETURN FROM DRIVER
143632      INTEGER LETREG                           % T-REG WHEN LAST ERROR RETURN FROM DRIVER
143633      INTEGER LEAREG                           % A-REG WHEN LAST ERROR RETURN FROM DRIVER
143634      INTEGER LEDREG                           % D-REG WHEN LAST ERROR RETURN FROM DRIVER
143635      INTEGER ECTRG,ECARG,ECDRG,ECXRG          % CALLING PROGRAMS REGISTERS WHEN ERROR OCCURES
143641      TRIPLE LETAD=LETREG,FECTREG=ECTRG
143641
143641      BUFDISC: X BONE 6; GO FDIFEIL                           % TIMEOUT
143643
143643      FDIFEIL: X BONE 4=:HSTAT=:LEHSTAT                % RETURN HERE WHEN ERROR
143646           TAD=:LETAD;=CTADR=;FECTREG; CXRG=;ECXRG
143653           HSTAT=:SHSTAT(CFLUN)
143656           A\/AERRB=;AERRB; MIN ERCNT; 0/\0
143662           GO CFLRG
143663
143663      FDIFIN: X=:HSTAT                                 % RETURN HERE WHEN FINISHED
143664           A:=X=:SHSTAT(CFLUN)
143667           GO CFLRG
143670
143670      RBUS
143700
143700      *"BDBUG
"143700
```

```
143700   %
143700   +8DLP1+8DLP2+8DVE1+8DVE2
"143700   %=============================================================================
143700   % 39.7        T L P R I N T    C L P 1 0
143700   %
143700   %
143700   % ROUTINE ON LEVEL 11 TO START DATA TRANSFER TO THE DEVICE ON MONITOR LEVEL
143700   %
143700   SUBR TLPRINT,CLP10
143700
143700   % VARIABLES IN MASS STORAGE DATAFIELD
143700   DISP -41
143700   INTEGER VEFLG                  % VEFLG=1 IF VERSATEC ELSE VEFLG=0
143700   INTEGER VEMOD                  % FUNCTION CODE IF NOT WRITE ON VERSATEC
143700   PSID
143700   DISP -17
143700   INTEGER CIOXC                  % CODE TO SET OUT IN IOX-WRITE-CONTROL
143700   INTEGER IOXWC                  % CODE FOR IOX-WRITE-CONTROL
143700   INTEGER IOXRS                  % CODE FOR IOX-READ-STATUS
143700   INTEGER IOXWD(2)               % CODE FOR IOX-WRITE-DATA
143700   INTEGER LP5MF                  % MONITOR FUNCTION ROUTINE ADDRESS
143700   INTEGER CBHOLD                 % NUMBER OF BYTES IN DEVICE BUFFER
143700   INTEGER CHENTE                 % BYTE NUMBER TO FETCH FROM BUFFER
143700   INTEGER CBUFST                 % DEVICE BUFFER ADDRESS
143700   PSID
143700
143700   % VARIABLES IN I/O DATAFIELD                                          --
143700                                        % I/O DATAFIELD (ONLY VERSATEC)
143700   DISP 0; DOUBLE POINTER DP1=P1; PSID
143700
143700   TLPRINT: X:=:B
143701           CALL PICKFPAR;
143702           T=:X.TRG; AD=:X.MEMAD; A:=D=:X.CBUFST
143706           CALL PICKLPAR; A=:X.DRG; T=:X.XRG
143711           X=:B; 0=:HSTAT
143713   %  %%% IF DRG><0 THEN 173; GO FINER FI % UNNECESSARY TEST
143713           IF TRG=1 THEN
143717                 IF XRG=0 GO CRTACT              % NO WORDS TO TRANSFER
143721                 A SH 1=:CBHOLD; 0=:CHENTE
143724   FELS:         "LP5MF"-"MFUNC"+B=:B
143730   CRTACT:       CALL RTACT; CALL WT11; GO ERR22
143733           FI
143733           IF A=30 OR A=31 OR A=32 THEN
143744                 IF VEFLG><0 THEN
143746                    T=:VEMOD; 0=:CBHOLD; GO FELS
143751                 FI
143751           FI; 201
143752   FINER:   A=:CERRCODE; 20=:HSTAT; GO CRTACT
143756
143756   % ENTRY POINT ON LEVEL 10 AFTER EACH INTERRUPT
143756   CLP10: B=:X; "LP5MF"-"MFUNC"+B=:B; CALL RTACT; X=:B; CALL ID10; GO CLP10
143767
143767   RBUS
144001
144001
144001   %=============================================================================
144001   % 39.8        X T R D L P
144001   %
144001   %
144001   % SUBROUTINE ON MONITOR LEVEL TO TRANSFER DATA TO THE DEVICE
```

```
144001    %
144001
144001    SUBR XTRDLP
144001
144001    INTEGER WDIO1=?,RSIO1=?,WDIO2=?,RSIO2=?
144001
144001    % VARIABLES IN MASS STORAGE DATAFIELD
144001    DISP -41
144001    INTEGER VEFLG                 % FLAG=1 IF VERSATEC ELSE EQUALS ZERO
144001    INTEGER VEMOD                 % FUNCTION CODE IF NOT WRITE ON VERSATEC
144001    INTEGER POINTER CHCONV        % ADDRESS OF CHARACTER CONVERTING ROUTINE
144001    PSID
144001                                          % IN I/O DATAFIELD
144001    DISP -24
144001    INTEGER CXMAX                 % MAXIMUM OF BYTES IN EXTRA-BUFFER
144001    INTEGER CXBHOLD               % ACTUAL NUMBER OF BYTES IN EXTRA-BUFFER
144001    INTEGER CXBUFST               % ADDRESS OF EXTRA-BUFFER
144001    INTEGER CXHENTE(2)            % HENTE-POINTER IN EXTRA-BUFFER
144001    INTEGER CIOXC                 % CODE TO SET OUT IN IOX-WRITE-CONTROL
144001    INTEGER IOXWC                 % CODE FOR IOX-WRITE-CONTROL
144001    INTEGER IOXRS                 % CODE FOR IOX-READ-STATUS
144001    INTEGER IOXWD(2)              % CODE FOR IOX-WRITE-DATA
144001    INTEGER LP5MF                 % ADDRESS OF MONITOR FUNCTION ROUTINE
144001    INTEGER CBHOLD                % CURRENT NUMBER OF BYTES IN DEVICE BUFFER
144001    INTEGER CHENTE                % BYTE NUMBER TO FETCH FROM BUFFER
144001    INTEGER CBUFST                % DEVICE BUFFER ADDRESS
144001    PSID
144001
144001    XTRDLP: X.S0=:B; 0=:TMR
144004            IF VEFLG><0 THEN
144006                IF PVEFUNC=10 THEN 104 ELSE 4 FI; A=:CIOXC
144016            FI
144016            IF CXBHOLD><0 THEN              % NUMBER OF BYTES IN EXTRA BUFFER
144020               X:=CXHENTE; GO L1
144022            FI
144022    L0:     IF CBHOLD=0 GO FIN              % NUMBER OF BYTES IN DEVICE BUFFER
144024    @LIB CXCPU-,
144024    @LIB CXCPU
144024            IF A<<=CXMAX THEN
144027               % COPY FROM DEVICE BUFFER/ABSTR BUFFER TO EXTRA BUFFER IN DATAFIELD
144027               A:=CHENTE SHZ -1+CBUFST=:D; A:=CBHOLD=:CXBHOLD+1 SHZ -1=:L
144040               A:=MEMA1; T:=CXBUFST; X:=0; *MOVPP
144044    @ELIB
144044                    IF VEMOD=0 THEN
144046                      IF RTRES><0 THEN CALL RTACT FI
144051                    FI
144051                    IF CHENTE BIT "0" THEN X:=1 ELSE X:=0 FI
144057                    0=:CBHOLD                       % ODD BYTE LEFT OVER ?
144060    L1:             IOXRS=:RSIO1; IOXWD=:WDIO1
144064                    DO
144064                         INTEGER RSIO1        % READ STATUS
144065                    WHILE A BIT 3
144067                         T:=CXBUFST
144070                         IF CHCONV><0 THEN
144072                            *LBYT
144073                            CALL CONVCH
144074                         ELSE
144075                            *LBYT
144076                         FI
144076                         INTEGER WDIO1
```

```
144077                      T:=IOXWC; CIOXC; *EXR ST
144102                      X=:A; FOR X:=-5 DO OD; X:=A                % DELAY
144106                      X+1; CXBHOLD-1=:CXBHOLD
144112                      IF A=0 THEN
144113                          IF CBHOLD=0 GO PIN1
144115                          GO L0
144116                      FI
144116                  OD
144117  PIN1:           X=:CXHENTE; T:=IOXWC; CIOXC BZERO 2 BONE "0"; *EXR ST
144125                  TTMR=:TMR; GO PSTUPR
144130  *)FILL
144133
144133  FIN:            IF VEMOD><0 THEN
144135                     0=:VEMOD
144136                     IF A=30 THEN
144141                        11
144142                     ELSE
144143                        IF A=31 THEN
144146                           10
144147                        ELSE
144150                           GO L2
144151                     FI;FI
144151                     A=:PVEFUNC; GO FINX
144153
144153  L2:                1505; T:=IOXWC; *EXR ST
144156                     GO PSTUPR
144157                  FI
144157  FINX:           IF RTRES><0 THEN CALL RTACT FI
144162                  GO PSTUPR
144163          FI
144163          IOXRS=:RSIO2; IOXWD=:WDIO2
144167          DO
144167              INTEGER RSIO2                                      % READ STATUS
144170          WHILE A BIT 3                                          % UNTIL NOT READY
144172              T:=MEMA1; A:=CHENTE=:D SHZ -1+CBUFST=:X
144200              *LDATX                                             % FETCH WORD FROM BUFFER
144201              IF D NBIT "0" THEN A SHZ -10 ELSE A/\377 FI        % GET BYTE
144206              MIN CHENTE
144207              IF T := CHCONV >< 0 THEN
144212                 CALL CONVCH                                     % CONVERT BYTE?
144213              FI
144213              INTEGER WDIO2                                      % WRITE DATA
144214              T:=IOXWC; CIOXC; *EXR ST
144217              X=:A; FOR X:=-5 DO OD; X:=A                        % DELAY
144223              CBHOLD-1=:CBHOLD
144226              IF A=0 GO PIN2
144227          OD
144230  PIN2:   T:=IOXWC; CIOXC BZERO 2 BONE "0"; *EXR ST
144235          TTMR=:TMR; GO PSTUPR
144240
144240  CONVCH:
144240          A=:D
144241          IF VEFLG><0 AND PVEFUNC=10 THEN A:=D ; EXIT FI
144251          A:=D/\177
144253          X=:D:="CHCONV"
144255          DO
144255                IF T:=X.S0=0 THEN X:=D; EXIT FI
144262                IF A=T THEN X.S1; X:=D; EXIT FI
144267                X+2
144270          OD
```

```
144271    RBUS
144277
144277
144277
144277    *"8HDMA
144277
144277    @LIB CXCPU-,
144277    @LIB CXCPU
144277    %=============================================================================
144277    %       R E T I N D V   -   M 8 8 I W O R D
144277    %
144277    SUBR M8IDRET,M88IWORD
144277
144277    DISP -1; INTEGER MN1; PSID
144277
144277    NOSPACE: DFOPP=:B; GO WDX
144302    @ELIB
144302    M8IDRET:
144302    RETINDV: IF X:=FYLLE=0 THEN X:=MAX FI; X-1; T:=BUFST; *LBYT
144311    INDRET: A=:CHARI; DFOPP=:B; CALL STDEV; GO MOURET
144316
144316    M88IWORD:  DFOPP=:B; CFREE=:D; IF A-4<0 GO NOSPACE
144324           BUFST+FYLLE=:X; MAX+BUFST=:L; *IRR ALEVB DA
144333           IF A=0 THEN DFOPP=:B; GO MOURET FI
144337           A=:X.S0; IF X+1=L THEN X:=BUFST FI; D-1; MIN BHOLD; *IRR ALEVB DD
144347           IF A=0 GO NOMORE; A=:X.S0; IF X+1=L THEN X:=BUFST FI
144355           D-1; MIN BHOLD; *IRR ALEVB DL
144360           IF A=0 GO NOMORE; A=:X.S0; IF X+1=L THEN X:=BUFST FI
144366           D-1; MIN BHOLD; *IRR ALEVB DX
144371           IF A=0 GO NOMORE; A=:X.S0; IF X+1=L THEN X:=BUFST FI
144377    NOMORE:  A:=X-BUFST=:FYLLE; A:=D=:CFREE
144404           IF X=BUFST THEN X:=L FI; X.MN1; GO INDRET
144412
144412    RBUS
144414
144414    @LIB CXCPU-,
144414    @LIB CXCPU
144414    %=============================================================================
144414    %       B 8 I W O R D   -
144414    %
144414    SUBR B8IWORD
144414
144414    NOSPACE: DFOPP=:B; GO WDX
144417    @ELIB
144417
144417    B8IWORD: DFOPP=:B; IF CFREE-4<0 GO NOSPACE
144424           BUFST+FYLLE=:X; BUFST+MAX=:L
144432           *IRR ALEVB DA
144433           A=:X.S0; IF X+1=L THEN X:=BUFST FI; *IRR ALEVB DD
144441           A=:X.S0; IF X+1=L THEN X:=BUFST FI; *IRR ALEVB DL
144447           A=:X.S0; IF X+1=L THEN X:=BUFST FI; *IRR ALEVB DX
144455           A=:X.S0; IF X+1=L THEN X:=BUFST FI
144462           BHOLD+4=:BHOLD; CFREE-4=:CFREE; A:=X-BUFST=:FYLLE
144473           *IRR ALEVB DX
144474    RETINDV: A=:CHARI; DFOPP=:B; CALL STDEV; GO MOURET
144501
144501    RBUS
144503
144503    @LIB CXCPU-,
144503
```

```
144503    SUBR NN310
144503    INTEGER CCHAR,BR310
144505    BRCNN: IF DFLAG BIT 5SPEC THEN 1=:BR310; EXIT FI; EXITA
144514    NN310: 0=:BR310; *POF
144516           CALL IOTRANS; GO WDX; A SHZ 10=:CCHAR; *IRW ALEVB DA;
144523           X:=1; CALL BRCNN; GO GORET
144526           CALL IOTRANS; GO GORET; A\/CCHAR; *IRW ALEVB DA;
144532           X+1; CALL BRCNN; GO GORET
144535           CALL IOTRANS; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DD;
144542           X+1; CALL BRCNN; GO GORET
144545           CALL IOTRANS; GO GORET; A \/CCHAR; *IRW ALEVB DD;
144551           X+1; CALL BRCNN; GO GORET
144554           CALL IOTRANS; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DL;
144561           X+1; CALL BRCNN; GO GORET
144564           CALL IOTRANS; GO GORET; A\/CCHAR; *IRW ALEVB DL;
144570           X+1; CALL BRCNN; GO GORET
144573           CALL IOTRANS; GO GORET; A SHZ 10=:CCHAR; *IRW ALEVB DX;
144600           X+1; CALL BRCNN; GO GORET
144603           CALL IOTRANS; GO GORET; A\/CCHAR; *IRW ALEVB DX;
144607           X+1; CALL BRCNN; GO GORET
144612    GORET: A:=X; IF X:=BR310><0 THEN A BONE 17 FI; *IRW ALEVB DT
144617           GO MOURET
144620    RBUS
144622
144622    @LIB CXCPU-,
144622    @LIB CXCPU
144622    %=============================================================================
144622    %       B 8 I I N D V   -   M 8 I I N D V
144622    %
144622    SUBR B8IINDV,M8IINDV
144622    @ELIB
144622
144622    B8IINDV:
144622    M8IINDV: IF BHOLD=0 GO WDX
144625           IF "IOTRANS"="IGTCH" GO IB8INTERM
144631           BHOLD=:D; MAX+BUFST=:L; HENTE+BUFST=:X
144641           X.S0; *IRW ALEVB DA; ION
144644           T:=1; IF D-1=0 GO IRET; IF X+1=L THEN X:=BUFST FI; X.S0; *IRW ALEVB DD
144656           T:=2; IF D-1=0 GO IRET; IF X+1=L THEN X:=BUFST FI; X.S0; *IRW ALEVB DL
144670           T:=3; IF D-1=0 GO IRET; IF X+1=L THEN X:=BUFST FI; X.S0; *IRW ALEVB DX
144702           D-1; T:=4
144704    IRET:  A:=D=:BHOLD; A:=T; *IRW ALEVB DT
144710           CFREE+T=:CFREE; IF X+1=L THEN X:=BUFST FI; A:=X-BUFST=:HENTE
144722           GO MOURET
144723
144723    RBUS
144727
144727    *"8VIPS
"144727    @ELIB
144727    @LIB CXCPU-,
144727    @LIB CXCPU
144727    %=============================================================================
144727    %       B 4 I I N D V
144727    %
144727    SUBR B4IINDV
144727    @ELIB
144727
144727    B4IINDV: IF BHOLD<4 GO WDX
144733           IF "IOTRANS"="IGTCH" GO IB4INTERM
144737           MAX+BUFST=:L; HENTE+BUFST=:X
```

```
 44745           X.SO; *IRW ALEVB DA
 44747           IF X+1=L THEN X:=BUFST FI; X.SO; *IRW ALEVB DD
 44755           IF X+1=L THEN X:=BUFST FI; X.SO; *IRW ALEVB DL
 44763           IF X+1=L THEN X:=BUFST FI; X.SO; *IRW ALEVB DX
 44771           T:=4
 44772   WORET:  BHOLD-T=:BHOLD; A:=T; *IRW ALEVB DT
 44777           CFREE+T=:CFREE; IF X+1=L THEN X:=BUFST FI; A:=X-BUFST=:HENTE
 45011           *ION
 45012           GO MOURET
 45013
 45013   RBUS
 45017
 45017
 45017   * 8CXHD+8C1X2+8C2X2+8C3X2+8C4X2
```

```
"145017
145017   %=====================================================================
145017   % 39.12
145017   %      H D L C    A C T 1 2    A C T 1 3
145017   %      HDLC -  NONE-RESIDENT PART
145017   %
145017   % ENTRY:  B-REG POINTS TO DATAFIELD
145017   %
145017   %         POF, ION, LEV 3 DISABLED
145017   %
145017   INTEGER POLSZ=?
145017   SUBR HDLC,ACT12,ACT13,IMHDLC
145017   %
145017   DISP -1; INTEGER BCOMT; PSID
145017   INTEGER MESSM:=0                          %INCREMENTED FOR EACH SEND
145020   HDLC:  A:=L=:"RSRET"                      %SAVE RETURN ADDRESS
145022          IF HINIF >< -1 THEN                % POOL INITIATED ?
145026            MASTB =: XXSBK                   % BANKNO TO COPY ROUTINE
145030            MAX -1 SHZ -1 + BUFST =: D; A:=BUFST
145036            X:=B; T:=MASTB; *AAX XBBNK
145041            CALL ZBINI; CALL PZCRA; -1=:HINIF       %MARK AS INITIATED
145045          FI
145045          IF DDD0 = FSEND THEN
145051   %
145051   % SEND
145051   %
145051             IF DDD3 < 0 THEN T:=EMSGS; GO RSRET FI     %ILLEGAL MESSAGE SIZE
145055             IF A > DDD4 THEN T:=EMAXS; GO RSRET FI     % MAX SIZE < USED SIZE
145062             X:="XBBNK"+B; "BHEAD+BHEAD"+DDD4; CALL ZBGET; CALL ZCRAS  %ALLOCATE BUFFER
145070             IF A = 0 THEN T:=ENBUF; GO RSRET FI
145073             A=:RSCUR=:X                              %USER PART OF MESSAGE
145075             IF MESSM + 1 < 0 THEN 1=:MESSM FI        %ONLY POSITIVE VALUES
145102             A=:MESSM=:MESSID; T:=MASTB; * BBID@3 STATX
145106             DDD4; * BMBYT@3 STATX                    %MAX BYTECOUNT IN MESSAGE
145110             DDD3; * BBYTC@3 STATX
145112             "OCHAIN"; *XCAHI@3 STATX                 % NOT XMSG DCB
145114             DDD2=:XXUBF; T:=DDD3=:D; X+BHEAD=:XXSBF; *AAB XXUBF
145123             CALL ZOPHY; * AAB -XXUBF; IOF            % COPY USER DATA TO BUFFER% SET MESSAGE IN QUEUE
145126             X:=RSCUR; CALL ICHAIN; *ION              % SET MESSAGE IN QUEUE
145131             CALL STDEV                               % ACTIVATE DRIVER
145132             0=:T; GO RSRET;
145134          FI
145134   %
145134   % RECEIVE
145134   %
145134   REMSG: *IOF
145135          CALL DOCHAIN; GO CHEMTY                  %REMOVE MESSAGE FROM QUEUE
145137          *ION
145140          X=:RSCUR; T:=MASTB; * BBID@3 LDATX
145143          A=:MESSID                                %MESSID OF CURRENT MESSAGE
145144          * BBYTC@3 LDATX
145145          A=:T
145146          IF DDD0-FMXRECV=0 AND DDD3<<T THEN
145154             T:=EMSGS=:DDD0; T:=A                  % ERROR, DCB GREATHER THAN SPECIFIED IN PARAMETER #4
145157          ELSE
145160             0=:DDD0
145161          FI; CALL STUSP;                          %BYTECOUNT TO USER
145162   IMHDLC:                                         % STUSP GOES HERE
145162          DDD2=:XXUBF; T=:D
145165          X:=RSCUR; X+BHEAD=:XXSBF; * AAB XXUBF
```

```
 45171          CALL ZOUSR
 45172          * AAB -XXUBF; COPY SB DX; AAX XBBNK
 45175          RSCUR; CALL ZBREL; CALL ZCRAS              %RELEASE BUFFER
 45200          T:=DDDO; GO RSRET                          %GO BACK TO RESIDENT PART
 45202    *)FILL
 45216    %
 45216    %
 45216    CHEMT: DDD4=:XWAITF
 45220          IF A = 1 THEN
 45223                X:=CURPR; CALL WDATA
 45225                *ION
 45226                B+OFSET; ZPREG-1=:ZPREG
 45232                CALL RETSTUPR
 45233          ELSE
 45234                *ION
 45235                T:=EEMTY; GO RSRET
 45237          FI
 45237          GO RSRET
 45240    %
 45240    PZCRA: IF A = 2 THEN T:=ENBUF; GO RSRET; FI
 45245          CALL ZCRAS
 45246    %
 45246    %     ROUTINES FOR ACTIVATION OF DRIVER PART
 45246    %
 45246    ACT12: IF WAKEF = 1 THEN EXIT FI
 45253          T:="STDRIV"; D:=L; CALL STL12; D=:P        %*80B*
 45257    %
 45257    %
 45257    ACT13: IF WAKEF = 1 THEN EXIT FI
 45264          T:="STDRIV"; D:=L; CALL STL13; D=:P                %*80B*
 45270    RBUS
```

```
145276
145276  %=======================================================================
145276  % 39.14     I C H A I N    O C H A I N
145276  %       HDLC - CHAINING ROUTINES
145276  %
145276  %
145276  %PURPOSE: ENTER OR REMOVE ENTRIES IN THE DRIVER QUEUE
145276  %
145276  %METHOD:      THE QUEUES ARE ARRANGED AS FIFOS.
145276  %             THE LINKING IS ONE WAY ONLY.
145276  %             0 IN THE LINKWORD IS USED T MARK END OF QUEUE.
145276  %             THE HEAD OF THE QUEUES IS IN THE APPROPRIATE DATAFIELD.
145276  %             IQUEU FOR INPUT, AND OQUEU FOR OUTPUT.
145276  %
145276  %             CHANGED FOR I VERSION:
145276  %             THE QUEUES MAY NOW APPEAR OUTSIDE BANK 0
145276  %             THE "MASTB" LOCATION IN DATAFIELD HOLDS THE BANK NUMBER
145276  %
145276  %ICHAIN: SET MESSAGE IN INPUT QUEUE (TO DRIVER)
145276  %OCHAIN: SET MESSAGE IN OUTPUT QUEUE (FROM DRIVER)
145276  %
145276  %ENTRY:  B-REG   DATAFIELDPOINTER
145276  %        X-REG   MESSAGEPOINTER (ICHAIN ONLY)
145276  %
145276  %EXIT:   B-REG   DATAFIELDPOINTER
145276  %        X-REG   MESSAGEPOINTER (OCHAIN ONLY)
145276  %
145276  SUBR ICHAIN,OCHAIN
145276  ICHAIN: IF IQUEU =0 THEN                                  % AM I ALONE ?
145300            X=:IQUEU; T:=MASTB; * BCHAI@3 STATX             % YES, 0 IN NEXT POINTER
145303            EXIT
145304          FI
145304          GO ICOC                                           % NO, LINK IN AT END OF LIST
145305  OCHAIN: T:=MASTB; * XCHAI@3 LDATX
145307          IF A >< "OCHAIN" THEN                             % SPECIAL CHAINING ROUTINE *81F*
145312            IF A >> XBSTR AND A << XBEND THEN EXIT FI        % CHECK POINTER
145321            *ION; COPY SA DP                                % RETURN TO "CALL OCHAIN"  *81F*
145323          FI
145323          IF ISTATE >< 0  THEN L=:D; CALL RTACT; D=:L; FI
145330          IF OQUEU =0 THEN                                  % AM I ALONE?
145332            X=:OQUEU; T:=MASTB; * BCHAI@3 STATX             % YES, 0 IN LINK WORD
145335            EXIT
145336          FI
145336  ICOC: X=:D:=A; T:=MASTB
145341     DO WHILE A >< 0
145342          A=:X; * BCHAI@3 LDATX
145344     OD
145345          A:=D; * BCHAI@3 STATX                  % LINK IN AT END OF LIST
145347          A=:X:=0; * BCHAI@3 STATX               % 0 AT END OF LIST
145352          EXIT
145353  RBUS
145355
145355
145355  %=======================================================================
145355  % 39.15     D I C H A I N    D O C H A I N
145355  %
145355  %DICHAIN: REMOVE OLDEST ENTRY FROM INPUT QUEUE (TO DRIVER)
145355  %DOCHAIN: REMOVE OLDEST ENTRY FROM OUTPUT QUEUE (FROM DRIVER)
145355  %TSTCH:   ANY MESSAGE IN DRIVER QUEUE.
145355  %
```

```
145355    %ENTRY:   B-REG  DATAFIELDPOINTER
145355    %
145355    %EXIT:    B-REG  DATAFIELDPOINTER
145355    %         X-REG  MESSAGEPOINTER
145355    %
145355    %         SKIP RETURN - MESSAGE FOUND
145355    %         NO SKIP     - EMTY QUEUE
145355    %
145355    SUBR DICHAIN,DOCHAIN,TSTCH
145355    TSTCH:    IF IQUEU><0 THEN EXITA FI EXIT
145361    DICHAIN:  IF IQUEU =0 THEN 0=:WAKEF; EXIT FI
145365              X:=A; T:=MASTB; * BCHAI@3 LDATX
145370              A =:IQUEU; 1=:WAKEF; EXITA;
145374    DOCHAIN:  IF OQUEU =0 THEN
145376                  IF DDD4 = 0 THEN -1=:ISTATE FI
145402                  EXIT
145403              FI
145403              X:=A; T:=MASTB; * BCHAI@3 LDATX
145406              A=:OQUEU; 0=:ISTATE; EXITA;
145411    RBUS
```

```
145411
145411   %=====================================================================
145411   % 39.16
145411   %       Z C R A S              INCONSISTENSY IN HDLC NONRESIDENT PART
145411   %
145411   %       T-REGISTER WILL CONTAIN OLD L-REGISTER WHEN STOP IN ERRFATAL
145411   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
145411   %
145411   SUBR ZCRAS
145411   INTEGER DHAAA                                    % ERRCODE
145412   INTEGER DHLLL                                    % WHERE I COME FROM
145413   ZCRAS: A=:DHAAA; A:=L=:DHLLL
145416          L=:T; A:=B+OFSET=:B; CALL 9ERR(#33)
145424          B-OFSET=:B; T:=HXERF; GO RSRET
145430   RBUS
```

```
145431
145431    *'
145431     *"8PIOC
145431     @LIB OLDPIOC
145431    *'
145431
145431    %-----------------------------------------------------------------------
145431    %       P R T C G P L O C
145431    %
145431    % SUBROUTINE TO THE RTCGPLOC IN RESIDENT
145431    % MUST BE CALLED IN PIOF.
145431    %
145431    SUBR PRTCGPLOC
145431    INTEGER TREG,AREG,DREG,CCDISP; TRIPLE SVTAD=TREG
145435    INTEGER ARRAY POINTER ACCTAB:=CCTAB
145436    PRTCGPLOC:
145436            TAD=:SVTAD; X=:XRG; A:=X/\1777=:CCDISP:=X SHZ -12+100=:D; X:=0
145450            DO WHILE ACCTAB(X)><-1
145454               IF A/\377=D THEN
145457                  X+1; A:=ACCTAB(X)=:D:=0; AD SH 12; X:=CCDISP
145465                  X+D; T:=A
145467                  IF TREG=0 THEN
145471                     *LDATX; STA AREG
145473                  ELSE
145474                     AREG; *STATX
145476                  FI; GO  OUT
145477               FI; X+2
145500            OD; L-1
145502    OUT:    TAD:=SVTAD; X:=XRG; EXITA
145505    RBUS
145507
145507
145507    @DEV 1
145507    @DEV (S-S-J)MRES-CDR3
145507
145507    %%%%%%%%%%%%%%%%%%%%%%%%%%%% M R E S  - C D R 3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
145507    %=======================================================================
145507    %% TABLE FOR CPU-TIME ACCOUNTING FOR RT-PROGRAMS
145507    *"8ACC
145507     *ACTAB=*
145507    **<*+8RTN+8RTN+8RTN
145507    *)ZERO
145507    **+8RTN+8RTN+8RTN/
145614    *"
145614     %%
145614    %=======================================================================
145614    %% TABLE FOR BLOCK-IO ACCOUNTING
145614    *"8IOAC
145614     *RTPNB=NXRTP+8RTN                  % TOTAL NUMBER OF RT-PROGRAM-DESCRIPTIONS
145614    *RTPNB:
000063  145614     *IOACT=*
145614    **+RTPNB+RTPNB/
145762    *'
```

```
"145762
145762   *"-8LOG; LOG1=TEXIT; LOG2=TEXIT; LGARR=0; LOGFI,0
"145762   %=============================================================================
145762   % 40.3        L O G 1    L O G 2    L G A R R
145762   %
145762   % ROUTINES TO GATHER SYSTEM AND PROGRAM STATISTICS
145762   % CALLED FROM TERMINAL OUTPUT DRIVER (LEVEL 10)
145762   % THE INFORMATION IS USED BY THE COMMAND "RT-PROGRAM-LOG"
145762
145762   INTEGER HHISTART=?,HTESTPROG=?,HFLGHISTO=?
145762   INTEGER COMRTP,ALLRTP
145764   INTEGER ARRAY LGARR(0)
145764   SUBR LOG1,XHISTI
145764   INTEGER ACTC1,ACTC2,ACTP1,ACTP2          % NOTE THAT THESE VARIABLES CORRESPOND
145770   INTEGER SWC1,SWC2,SWP1,SWP2              % TO SYMBOLS DECLAIRED ON THE OPCOM SEGM
145774   INTEGER FILC1,FILC2,FILP1,FILP2          % IN THE RT-PROGRAM-LOG COMMAND
146000   INTEGER DIC1,DIC2
146002   INTEGER PASP1,PASP2
146004   INTEGER IOWP1,IOWP2
146006   INTEGER TOTL1,TOTL2
146010   INTEGER U1T1,U1T2,U1P1,U1P2
146014   INTEGER U2T1,U2T2,U2P1,U2P2
146020   INTEGER ADRSWP,ADRU1,ADRU2
146023   INTEGER LOGFIELD,TESTPROG
146025   INTEGER PTOT1,PTOT2
146027   INTEGER INCRL,CINCR
146031
146031   INTEGER POINTER PTESTPROG=?
146031
146031   % SUBROUTINE TO GET STATISTICS
146031   % B=TERMINAL OUTPUT DATAFIELD
146031   % FOR VSX: X=ADDR OF RESIDENT DATAFIELD
146031   LOG1:
146031   @LIB CXCPU-,
146031   @LIB CXCPU
146031           IF LOGFIELD><X THEN EXIT FI
146035   @ELIB
146035           MIN CINCR; EXIT; INCRL=:CINCR  % INTERRUPTS BETWEEN SAMPLES
146041           IF COMRTP=0 GO TALLRTP
146043   @LIB CXCPU
146043           IF X.RTRES><"F1202".RTRES THEN
146050   @ELIB
146050   @LIB CXCPU-,
146050                   IF ALLRTP=0 THEN 0=:LOGFIELD FI; 0=:COMRTP
146054                   EXIT; *)FILL
146060           FI; *TRA PVL
146061           IF A SH 11 SHZ -14><0 THEN; *MIN ACTC2; SKP; MIN ACTC1; RAND
146070                   IF PTESTPROG=CURPROG THEN; *MIN ACTP2; SKP; MIN ACTP1; RAND
146100           FI      FI
146100           IF "CLFIE".RTRES><0 AND A><"1SWAP" THEN; *MIN SWC2; SKP; MIN SWC1; RAND
146112                   IF =PTESTPROG THEN; *MIN SWP2; SKP; MIN SWP1; RAND
146121           FI      FI; GO L1; *)FILL
146125   INTEGER POINTER PTESTPROG:=TESTPROG
146126   L1:     IF ADRSWP.RTRES=:D><0 THEN; *MIN DIC2; SKP; MIN DIC1; RAND
146136                   IF "CLFIE".RTRES=0 OR A="1SWAP" THEN; *MIN FILC2; SKP; MIN FILC1; RAND
146150                     IF D=PTESTPROG THEN; *MIN FILP2; SKP; MIN FILP1; RAND
146157           FI      FI FI
146157           IF PTESTPROG.WLINK=0 THEN; *MIN PASP2; SKP; MIN PASP1; RAND
146166           FI
146166           IF X.STATUS BIT 5WAIT THEN; *MIN IOWP2; SKP; MIN IOWP1
```

```
146174          FI
146174          *MIN TOTL2; SKP; MIN TOTL1; RAND
146200          IF ADRU1><0 THEN
146202                IF A.RTRES><0 THEN; *MIN I (U1T2; SKP; MIN I (U1T1; RAND
146211                   IF A=PTESTPROG THEN; *MIN I (U1P2; SKP; MIN I (U1P1; RAND
146220          FI    FI    FI
146220          IF ADRU2><0 THEN
146222                IF A.RTRES><0 THEN; *MIN I (U2T2; SKP; MIN I (U2T1; RAND
146231                   IF A=PTESTPROG THEN; *MIN I (U2P2; SKP; MIN I (U2P1; RAND
146240          FI    FI    FI
146240  TALLRTP: IF ALLRTP=0 THEN EXIT FI
146243  @LIB CXCPU-,
146243  @LIB CXCPU
146243          IF X.RTRES><"F1352".RTRES THEN EXIT FI
146251  @ELIB
146251          CURPROG-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
146257          A SH 1+"PGARR"=:X; *MIN 1,X; SKP; MIN ,X; RAND
146266          *MIN I IPTO2; SKP; MIN I IPTO1; RAND
146272  *"8LOG 8HIST
"146272           IF HTESTPROG=PTESTPROG AND HFLGHISTO><0 AND HHISTART=0 GO XHISTI
146303  *"8LOG
"146303           EXIT
146304  INTEGER POINTER IPTO1:=PTOT1,IPTO2:=PTOT2
146306  RBUS
146331  INTEGER ARRAY PGARR(0)
146331  * *+NXRTP+NXRTP+8RTN+8RTN+10/
146507  INTEGER ARRAY PGAEND(0)
146507  INTEGER ARRAY SLVTB(24)
146533  INTEGER LOTH1,LOTH2
146535
146535  *
146535  *"8HIST
```

```
"146535
146535   %=============================================================================
146535   % 40.4        H I S T I   H I S T O
146535   %
146535   % SUBROUTINE TO MAKE P-REGISTER HISTOGRAM FOR PROGRAMS
146535   % CALLED BY ICLK, MONITOR LEVEL
146535   % USED BY THE COMMAND "PRINT-HISTOGRAM"
146535
146535   INTEGER ARRAY HISTO(200)
146735   SUBR HISTI,SHISTI,XHISTI
146735   DOUBLE POUTSIDE; INTEGER TOTL1,TOTL2,STLT1,STLT2 % NOTE THAT THESE VARIABLES
146743   INTEGER TESTPROG,HISTART,FPREG,DELTA,HISTFLAG,CHILEV,FLGHIST % CORRESPOND TO
146752   INTEGER XREG                                % SYMBOLS DECLAIRED AT OPSEG IN THE
146753   *HHIST=HISTA; HTEST=TESTP; HFLGH=FLGHI % DEFINE-HISTOGRAM COMMAND
146753   HISTI: IF HISTART=0 OR HISTFLAG><0 OR CURPROG><TESTPROG THEN EXIT FI; X=:XREG
146765          IF BACKGROUND=0 OR CURPROG.ACTPRI/\3=0 THEN
146773                *IRR ALEVB DP
146774                A-FPREG=:D:=0; T:=DELTA; *RDIV ST
147001                IF A<<100 THEN X:="HISTO"+A+A ELSE X:="POUTSIDE" FI
147011                *MIN 1 ,X; SKP; MIN 0 ,X; RAND
147015          ELSE
147016                MIN STLT2; *SKP; MIN STLT1; RAND
147022          FI; MIN TOTL2; *SKP; MIN TOTL1; RAND
147026          X:=XREG; EXIT
147030   XHISTI: *TRA PVL
147031          A SHZ -3/\17=:D
147034          IF A>=12 THEN
147037                MIN LOTH2; *SKP
147041                MIN LOTH1; *RAND; EXIT
147044          FI
147044          A SH 1+"SLVTB"=:X
147047          *MIN 1,X; SKP; MIN ,X; RAND
147053          IF CHILEV SHZ -3/\17><D THEN EXIT FI
147061   SHISTI: CHILEV=:T; A SHZ -3/\17=:X; *EXR ST
147067          IF A=PVLAD(X) THEN T.S2 FI      % USE SAVED P-REG ON PVL
147074          A-FPREG=:D:=0; T:=DELTA; *RDIV ST
147101          IF A<<100 THEN X:="HISTO"+A+A ELSE X:="POUTSIDE" FI
147111          *MIN 1,X; SKP; MIN ,X; RAND
147115          *MIN TOTL2; SKP; MIN TOTL1; RAND
147121          EXIT
147122
147122   RBUS
147134   *"
"147134    +"7NDDI
```

```
147134
147134    %=============================================================================
147134    % 40.9
147134    %       H D L C  -  D R I V E R
147134    %       SYMBOLS AND DEFENITIONS
147134    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
147134    %
147134    %
147134    %       COMMON DEFINITIONS
147134
147134    SYMBOL INTCH = 102164                   % RESPONCE ON INIT
147134    SYMBOL SILFO = 100000                   % ILLEGAL TRANSMITTER LIST KEY
147134    SYMBOL TXUND =      2                   % TRANSMITTER UNDERRUN
147134    SYMBOL OVERR = 100000                   % RECEIVER OVERRUN
147134    SYMBOL EMTY  =  4000                    % RECEIVER LIST EMTY
147134    SYMBOL BLDON = 10                       % RECEIVER STATUS BLOCK DONE
147134    %
147134    %
147134    %       I O X DEFINITIONS
147134
147134    @ICR;
147134    SYMBOL RRDR=0,                          % READ REC.DATA REG.
147134           WPCR,                            % WRITE PARAM. CONTR. REG.
147134           RRS,                             % READ REC STATUS
147134           WSAR,                            % WRITE SYNC/ADDRESS REG.
147134           WCHL,                            % WRITE CHAR. LENGTH
147134           WTDR,                            % WRITE TRANSM. DATA REG.
147134           RTSR,                            % READ TRANSM. STATUS REG.
147134           WTCR,                            % WRITE TRANSM. CONTROLE REG.
147134           RRTS,                            % READ REC. TRANSFER STATUS
147134           WRTC,                            % WRITE REC. TRANSFER CONTROLE
147134           RTTS,                            % READ TRANSM. TRANSFER STATUS
147134           WTTC,                            % WRITE TRANSM. TRANSFER CONTROLE
147134           RDMA,                            % READ DMA ADDRESS (LEAST)
147134           WDMA,                            % WRITE DMA ADDRESS (LEAST)
147134           RDCR,                            % READ DMA COMMAND REG
147134           WDCR;                            % WRITE DMA COMMAND REG+TRIGGER
147134    @CR;
147134    %
147134
147134
147134    %       ERROR CODES SENT FOR USER
147134
147134    SYMBOL ENCLEAR = 100                    % DEV. NOT CLEARED BEFORE INITIATED
147134    SYMBOL ENINIT  = 101                    % DEV. NOT INITIATED BEFORE TRANSMITTION
147134    SYMBOL EUND    = 102                    % UNDERRUN
147134    SYMBOL ETOU1   = 103                    % TRANSFER TIMEOUT
147134    SYMBOL ETOU2   = 104                    % COMMAND TIMEOUT (LOCAL IN IOF)
147134    SYMBOL EILFNC  = 105                    % ILLEGAL FUNCTION
147134    SYMBOL EILLINT = 106                    % HW. FAILURE ININIT.
147134    SYMBOL ETOMUCH  = 107                   % RECEIVER LIST OVERFLOW
147134    SYMBOL ECLEAR   = 110                   % DEVICE CLEARED, MESSAGE RETURNED
147134    SYMBOL EINP     = 111                   % OVERRUN OR CRC ERROR ON INPUT
147134    SYMBOL EPAR     = 112                   % ILLEGAL PARAMETER
147134    SYMBOL EILSIZ   =113                    % BYTECOUNT (FRAMD DATA PART) > FRAMESIZE
147134    SYMBOL ETOSM    =114                    % TOO SMALL MESSAGE FOR EXPECTED INFO
147134    SYMBOL EILFZ    =115                    % ATTEMPT TO SEND FRAME < 2 BYTES
147134    SYMBOL EX21     =116                    % CONNECTION BROKEN BY X21
147134    SYMBOL EDISP    =117                    % ILLEGAL DISPLACEMENT SPESIFICATION
147134    SYMBOL ELOCK    =120                    % DRIVER LOCKED (POSSIBLY BY X21)
```

```
147134   %
147134
147134
147134   %       LIST DECRIPTION
147134
147134   DISP 0
147134           INTEGER LKEY                          % LIST KEY
147134           DOUBLE DLSTST=LKEY
147134           INTEGER LBYTC                         % BYTECOUNT
147134           INTEGER LMEM1                         % MOUST SIGNIFICANT MEMORY ADDRESS
147134           INTEGER LMEM2                         % LEAST SIGNIFICANT MEMORY ADDRESS
147134           DOUBLE LMEM = LMEM1                   % *81F*
147134           INTEGER LEOL                          % END OF LIST
147134   PSID
147134   %
147134   SYMBOL LMASK = 60377
147134   SYMBOL HX21M = 60000
147134   SYMBOL HX21S = 16
147134   SYMBOL HX21D = 15
147134   %
147134   %       LKEY VALUES
147134   %
147134   SYMBOL FSERM = 2003                           % FULL TRANSMITTER BLOCK
147134   SYMBOL FEOL  = 0                              % END OF LIST
147134   SYMBOL ERB   = 1000                           % EMTY RECEIVER BLOCK
147134   SYMBOL NLP   = 3000                           % NEW LIST POINTER
147134   SYMBOL XBLDN  =   10                          % BLOCK DONE
147134   %
147134
147134
147134   %       USER MESSAGE DECRIPTION
147134
147134   SYMBOL CHEAD = 3                              % LENGTH OF USER HEAD
147134   SYMBOL BCHEA = CHEAD+CHEAD                    % LENGTH OF USER HEAD IN BYTES
147134   DISP 0
147134           INTEGER XCFUNC                        % FUNCTION
147134           INTEGER CRET                          % RETURN STATUS INFO
147134           INTEGER ADSTA                         % ADDITIONAL STATUS INFO.
147134   %       LOCATIONS USED BY XMSG FUNCTIONS *81F*
147134           INTEGER XHBYT                         % BYTECOUNT OF RECEIVER FRAME
147134           DOUBLE XHBUF                          % RECEIVER DATA ADDRESS
147134           DOUBLE XMLIS = XHBYT                  % ADDRESS OF TRANSMITTER LIST
147134   PSID
147134   %
147134   %       XCFUNC VALUES
147134   %
147134   SYMBOL DATAM = 1                               % DATA TO BE TRANSFERED
147134   SYMBOL XRESET= 2                               % RESET THIS PORT
147134   SYMBOL MCLEAR=3                               % CLEAR INTERFACE
147134   SYMBOL INIT = 4                               % INITIATE INTERFACE
147134   SYMBOL PSTAT = 5                              % GET PORT STAUS
147134   SYMBOL DTLI = 6                               % DEFINE TRANSMITTER LIST * 81F*
147134   SYMBOL DRBI = 7                               % DECLARE RECEIVER BUFFER * 81F*
147134   %
147134
147134   %       MESSAGE DECRIPTIONS
147134   %
147134
147134   %       INIT
147134   %
```

```
147134    SYMBOL MAMOD = 2                    % MAINTENANCE MODUS
147134    SYMBOL HALF=1                       % HALF DUPLEX MODUS
147134    SYMBOL FDPLX=0                       % FULL DUPLEX MODUS
147134    %
147134    DISP 3
147134            INTEGER IMODUS              % HALF/FULL - DUPLEX
147134            INTEGER IFSIZE              % MAX REC. FRAMESIZE
147134            INTEGER IRTRY               % # OF RETRYES IF TRANSFER ERROR
147134            INTEGER IDISP               % DISPLACEMENT
147134    PSID
147134    %
147134
147134    %       PSTAT
147134    %
147134    %
147134    DISP 0
147134            INTEGER ERRNO               % # OF ERRORS SINCE LASR PSTAT
147134            INTEGER ORERR               % OR FUNCTION OF ALL ERRORS
147134            INTEGER LHAST               % LAST HARDWARE STATUS
147134            INTEGER SSTPC               % # OF RECEIVER STOPPS
147134            INTEGER MAXEM               % MAX # OF EMTY BUFFERS HELD BY RECExIVER
147134            INTEGER HHMAX               % BUFFERSPACE FOR THIS LDN  IN BYTES
147134            INTEGER HHEAD               % SYSTEM PART OF EACH DCB  IN BYTES
147134    PSID
147134
147134    %
147134
147134    %       LOCATIONS USED BY DRIVER
147134
147134    DISP DIHDL = HARAL                  % *81F*
147134            INTEGER RTDYN               % DYNAMIC RETRY COUNTER
147134            INTEGER HDERC               % # OF ERROR TRANSFERS
147134            INTEGER DSTAT               % OR FUNCTION OF BAD STATUS
147134            INTEGER STPCNT              % # OF RECEIVER STOP
147134            INTEGER POINTER WTADR       % ADDRESS OF WT12 OR WT13
147134            INTEGER POINTER SRRES       % ADDRESS OF RESET ROUTINE
147134            INTEGER POINTER SRDAT       % ADDRESS OF TRANSFER ROUTINE
147134            INTEGER ACTSW               % SET IF DRIVER RUNNING, INPUT ONLY(ALLWAYS 77 FOR OUTPUT) *81F*
147134            INTEGER MAINT               % MAINTENANCE MODUS
147134            INTEGER HASTAT              % HARDWARE STATUS
147134            INTEGER LISTP               % POINTER TO DMA LIST
147134            INTEGER LIINT               % LIST ELEMENT WAITING FOR INTERRUPT
147134            INTEGER LISTF               % NEXT FREE LIST ELEMENT
147134            INTEGER OMSG                % CURRENT MESSAGE
147134            INTEGER INTSTA              % 1 IF CLEARED, 2 IF INITIATED, ELSE 0
147134            INTEGER CMODI               % 40 IF HALFD, 0 IF FULL DUPLEX
147134            INTEGER XRETRY               % # OF RETRYES WANTED IF TRANS.ERR
147134            INTEGER XREENT
147134            INTEGER XINITA              % *+1
147134            INTEGER PCREG               % PARAMETER CONTROLE REG.
147134            INTEGER SAREG               % SYNC/ADDRESS REG
147134            INTEGER CLENG               % CHARACTER LENGTH
147134            INTEGER DISP1               % DISPLACEMENT 1
147134            INTEGER DISP2               % DISPLACEMENT 2
147134            INTEGER MAXR                % MAX RECEIVER BLOC LENGTH
147134            INTEGER CHECK               % CHECKSUM
147134            INTEGER LISTL               % NUMBER OF ENTRIES IN  RECEIVER LIST
147134            INTEGER DCBX                % FIRST ENTRY IN DCB LIST (ONE FOR EACH ENTRY IN THE DMA LIST)
147134    PSID
147134    %
```

```
147134
147134   %=====================================================================
147134   % 40.10
147134   %       H D S T A     L T O U T     B A C K       S B Y T C
147134   %       F A L S E     D R E R R     R B Y T C     R A C T B
147134   %
147134   %       H D L C - DRIVER.     COMMON PART
147134   %
147134   %---------------------------------------------------------------------
147134
147134   %PURPOSE:     TRANSMITT DATA, AND CONTROLE THE OUTPUT PART
147134   %             OF THE HDLC DAM-INTERFACE.
147134
147134   %ENTRY:       B-REG - DATAFIELDPOINTER
147134   %             MESSAGE:        FUNCTION
147134   %                             STATUS
147134   %                             APPROPRIATE ADDITIONAL INFO
147134
147134   %EXIT:        STAUTS IS GIVEN WITHIN THE MESSAGE SEND
147134   %             BACK TO USER.
147134
147134   %OPERATION:   THIS COMMON PART IS ACTIVATED EVERY TIME A MESSAGE IS
147134   %             SENT FOR ANY HDLC-DRIVER. AN ACTION-ROUTINE WILL BE
147134   %             ACTIVATED ACCORDING TO THE FUNCTION IN THE MESSAGE.
147134   %             THE RESULTS OF DRIVER OPERATIONS IS SENT BACK TO THE
147134   %             USER WITHIN THE SAME MESSAGE.
147134   %             THE USER WILL GET THE RESULTS BY DOING A RECEIVE-MESSAGE
147134   %             COMMAND ON THE SAME PORT AS THE ORIGINAL MESSAGE
147134   %             WAS SENT. HOWEVER, THE RESULTS MAY BE RECEIVED ASYNCHRONOUSLY
147134   %             WITH RESPECT TO THE SENDING OF MESSAGES.
147134   %
147134   %
147134   %ACTIVATION POINT
147134   %
147134   SUBR   HDSTA,LTOUT,BACKX,SBYTC,FALSE,DRERR,RBYTC,RACTB,XMPAT,SADTS,SCRET
147134   INTEGER ARRAY HDENT:=(SMCLEAR,SIDAT,SIRES,SMCLEAR,HDSIN,SPSTA,XSSND,XSHDR) % *H*
147144   HDSTA: CALL DICHAIN;                                  %WAIT FOR MESSAGE
147145   FALSE: CALL WTADR                                      %ENTRY IF FALSE INTERRUPT
147146          CALL RBYTC                                      %MESSAGE LENGTH IN T-REG
147147          IF A>=6 THEN GO MSOK FI
147153          IF A>=4 THEN ETOSM =: X.CRET FI                 %MSSAGE TOO SMALL
147160          CALL OCHAIN; GO HDSTA
147162   MSOK:  X + BHEAD =: OMSG                               %CURRENT MESSAGE POINTER
147164   %
147164   %       GO TO ACTIONROUTINE
147164   %
147164          IF HXDOK ><X21OP THEN ELOCK; GO BACKX FI    % CHECK IF LOCKED BY X21
147172          T:=MASTB; * XCFUN@3 LDATX
147174          IF X:=A >> DRBI THEN                % *81F*
147200   HDEFR:   EILFNC
147201             ELSE
147202           HDENT(X); X:=OMSG; A=:P
147205          FI
147205   %
147205   %       EXIT SEQUENCE
147205   %
147205   %       B-REG - DATAFIELDPOINTER
147205   %       OMSG - CURRENT MESSAGE
147205   %       A-REG - RETURN STATUS INFO
147205   %
```

```
147205   BACKX:  X:=OMSG; T:=MASTB; * CRET@3 STATX
147210           X-BHEAD; T:=6; CALL SBYTC
147213           CALL OCHAIN
147214           GO HDSTA
147215   %
147215   %
147215   SIDAT: GO SRDAT                                    % JMP THROUGH DATAFIELD
147216   SIRES: CALL SRREST; GO BACKX
147220
147220   %       ENTRY: X POINTS TO DMA LIST ENTRY
147220   %              T - FRAME LENGTH+DISP1 (IF XMSG) OR TOTAL DCB LENGTH (IF NOT XMSG)
147220   %       THE CORRESPONDING DCB ENTRY IS FOUND IN THE DCBLIST
147220   %       THE DCB LENGTH OR FRAME LENGTH IS SAVED IN THE DCB
147220   %       EXIT: X POINTS TO THE DCB (USER PART)
147220
147220   XMPAT: 0=:X.LBYTC; A:=X-LISTP SHZ -2=:X
147225           * AAX DCBX; LDX 0,X,B                      % NOW X POINTS TO DCB
147227           T=:D:=MASTB; X - BHEAD; * BBID@3 LDATX
147233           IF A < 0 THEN                             % IS IT AN XMSG DCB ?
147234             A:=D; X + BHEAD; *XHBYT@3 STATX          % YES, SAVE FRAME LENGTH + DISP1
147237               ELSE
147240             A:=D+BCHEA; * BBYTC@3 STATX             % NO, SAVE TOTAL DCB LENGTH
147243             X + BHEAD
147244           FI; EXIT
147245   %
147245   %       INTERNAL TIMOUT
147245   %
147245   LTOUT: *JPC LOUT; EXIT
147247   LOUT:  A:=ETOU2; 0=:OMSG.ADSTA; *ION
147253           GO BACKX
147254   %
147254   %       STORE BYTECOUNT IN MESSAGE
147254   %
147254   SBYTC: T=:A:=MASTB; * BBYTC@3 STATX
147257           EXIT
147260   %
147260   %       STORE HW STATUS
147260   SADTS: T:=MASTB; * ADSTA@3 STATX
147262           EXIT
147263   %
147263   %       STORE STATUS
147263   SCRET: T:=MASTB; * CRET@3 STATX
147265           EXIT
147266   %
147266   %
147266   %       READ ACTUAL BYTECOUNT OF MESSAGE
147266   %
147266   RACTB: T:=MASTB; * BBYTC@3 LDATX
147270           EXIT
147271   %
147271   %       READ MAX BYTECOUNT OF MESSAGE
147271   %
147271   RBYTC: T:=MASTB; * BMBYT@3 LDATX
147273           EXIT
147274   %
147274   %
147274   %       XSDATA ERROR ROUTINE
147274   %
147274   %ENTRY:        A-REG            HARDWARE STATUS
147274   %             B-REG            DATAFIELDPOINTER
```

```
147274   %             X-REG            MESSAGE ADDRESS
147274   %
147274   DRERR: A\/DSTAT =: DSTAT                              %OR HW-STATUS
147276          HDERC+1 =: HDERC                               %ERROR COUNTER
147301          MIN RTDYN; GO SRDAT                            %TRY AGAIN
147303          XRETRY=:RTDYN; EXIT                             %TO MANY ERRORS, GIVE UP
147306   RBUS
147316
147316   SUBR ZXCHK
147316   %
147316   %       ZXCHK - CHECK XMSG DCB AND INTERFACE INITIALISED
147316   %
147316   ZXCHK: T:=MASTB; X-BHEAD; *BBID@3 LDATX
147321          X+BHEAD
147322          IF A > 0 THEN EILFNC; GO BACKX  FI             % THIS FUNCTION IS ONLY FOR XMSG *H*
147326          IF INTSTA >< 2 THEN ENINIT; GO BACKX FI
147334          EXIT
147335   RBUS
```

```
147336
147336   %:=================================================================
147336   % 40.11     X S D A T A    H O I N T    O I N I T    P O F T O
147336
147336   %
147336   %PURPOSE: TRANSMITT DATA TO REMOTE MACHINE
147336   %
147336   %OPERATION:  RECEIVES MESSAGES CONTAINING DATA TO BE SENT TO
147336   %            REMOTE MACHINE. THE RESULTS OF THE TRANSFER IS SENT
147336   %            BACK TO USER IN THE SAME MESSAGE.
147336   %
147336   %MESSAGE DECRIPTION:
147336   %          - FUNCTION = DATA
147336   %          - STATUS   = RET-STATUS
147336   %          - ADDSTA   = HARDWARE STATUS
147336   %          - DATA     = DATA TO BE TRANSMITTED / DMA LIST POINTER *81F*
147336   %
147336   SUBR XSDATA,HOINT,OINIT,POFTO,XSSND
147336   XSDATA: IF INTSTA >< 2 THEN A:=ENINIT; GO BACKX FI
147344           LISTP =: LIINT                              % *81F*
147346           X-BHEAD; CALL RACTB; X+BHEAD                %MESSAGE SIZE
147351           IF A < 7 THEN A:=EILFZ; GO BACKX FI          %MESSAGE IS TOO SMALL
147356           IF A-BCHEA>MAXR THEN A:=EILSIZ; GO BACKX FI  %TOO LONG FRAME
147364           A-DISP1=:LIINT.LBYTC                        %BYTECOUNT FOR LIST
147367           A:=OMSG+CHEAD=:X.LMEM2                      %SET BUFFER ADDR.
147372           MASTB=:X.LMEM1                              % PHYSICAL BANK
147374           FSERM=:X.LKEY                               %TRANSMITT ONE BLOCK ONLY
147376           0=:D                                        % *81F*
147377   %
147377   %       START INTERFACE
147377   %
147377   XHMST:  LIINT; T:=HDEV+WDMA; *IOF; EXR ST           %LIST ADDRESS *81F*
147404           A:=2000\/D; T+"WDCR-WDMA"; *EXR ST          %START TRANSMITTER  *81F*
147410           1+"RDCR-WDCR"; X:=-20;*EXR ST
147413           CALL LTOUT; *JAF *-2; ION
147416           1134+CMODI; T:=HDEV+WTTC; *EXR ST
147423           1 =: ACTSW                                  % *H*
147425           OMSG =: DCBX
147427   %
147427   %       ENABLE TIMEOUT
147427   %
147427   CONT:  A:=TTMR=:TMR
147431           CALL ID12                                   %WAIT FOR INTERRUPT
147432           GO HOINT; *)FILL
147443
147443   %
147443   %       OUTPUT INTERRUPT DETECTED,
147443   %       GET STATUS
147443   %       LOG STATUS, DEVICE NUMBER AND 1.WORD IN FRAME
147443   %       SEND FRAME TO USER
147443
147443           DISP 0; INTEGER AC; PSID
147443           INTEGER TELL:=0
147444           INTEGER DUIN:=0                             % INCREMENTED IF DUMMY INTERRUPTS *H*
147445           SYMBOL BUFSIZ=20
147445           SYMBOL TELMA=BUFSIZ-1
147445           INTEGER ARRAY BUFF0(BUFSIZ)                 % 1.WORD IN FRAME
147465           INTEGER ARRAY BUFF1(BUFSIZ)                 % DEVICE NUMBER
147505           INTEGER ARRAY BUFF2(BUFSIZ)                 % DEVICE STATUS
147525
```

```
147525   %
147525   HOINT: O=:TMR                                        %RESET TIMER
147526          T:=HDEV+RTTS; *EXR ST                         %READ STATUS
147531          A=:HASTAT                                     %SAVE STATUS
147532          IF T:=ACTSW = 0 THEN MIN DUIN; P+0; CALL WT12 FI          % *H*
147540          O=: ACTSW                                     % *H*
147541          IF CMODI = 40 THEN
147545            T:=HDEV+WTTC; *EXR ST    %TURN OFF RQTS
147550          FI
147550          MIN TELL; P+0
147552          TELL /\ TELMA=:TELL
147555          LIINT.LMEM2.AC=:BUFF0(TELL)                   % LOG 1.WORD IN FRAME  (BANK 0 ONLY) *81F*
147562          HDEV=:BUFF1(TELL)                             % LOG DEVICE USED
147565          HASTAT=:BUFF2(TELL)                           % LOG DEVICE STATUS
147570          X:=OMSG
147571          IF A/\ "SILFO+TXUND" = 0 THEN
147573                XRETRY=:RTDYN; A:=0; CALL SADTS
147577          ELSE
147600                A:=HASTAT; CALL SADTS; CALL DRERR
147603                A:=EUND
147604          FI
147604          O=:DCBX; GO FAR BACKX
147606   %
147606   %
147606   %       OUTPUT TIMEOUT ROUTINE
147606   %
147606   POFTO: X:=OMSG; O=:DCBX
147610          A:=CMODI; T:=HDEV+WTTC; *EXR ST
147614          A:=0; CALL SADTS; A:=ETOU1; GO FAR BACKX
147620   %
147620   %       SPECIAL ENTRY FOR PRIVILEGED USERS (MESSID<0)       *81F*
147620   %       THE FIRST TWO DATA WORDS OF THE DCB POINTS TO A
147620   %       TRANSMITTER LIST. THE LIST IS USED DIRECTLY BY HARDWARE.
147620   %
147620   XSSND: CALL ZXCHK                                    % CHECK DCB IS FROM XMSG
147621          T:=MASTB; *XMLIS@3 LDDTX
147623          A:=:D=:LIINT                                  % BANK BITS IN D-REG
147625          GO FAR XHMST
147626   RBUS
```

```
147641
147641    %=============================================================================
147641    % 40.12      H D R E C    H I I N T    S H D R E
147641    %
147641    %PURPOSE:     RECEIVE DATA FROM REMOTE MACHINE
147641    %
147641    %OPERATION:   RECEIVES EMTY (NO DATA) MESSAGES, WHICH WILL BE LINKED
147641    %              TO THE HDLC RECEIVER LIST. WHEN DATA (FRAMES) IS RECEIVED
147641    %              FROM REMOTE MACHINE, THE MESSAGES ARE FILLED AND SENT
147641    %              BACK TO USER.
147641    %MESSAGE DECRIPTION:
147641    %              - FUNCTION = EDATA
147641    %              - STATUS   = RET-STATUS
147641    %              - ADDSTA   = HARDWARE STATUS
147641    %              - DATA     = RECEIVED DATA
147641    %
147641    SUBR HDREC,HIINT,XSHDR
147641    HDREC: IF INTSTA >< 2 THEN A:=ENINIT; GO FAR BACKX FI
147647           X-BHEAD; CALL RBYTC; X+BHEAD; A-BCHEA      % GET MAX BYTECOUNT
147653           IF A < MAXR THEN A:=ETOSM; GO FAR BACKX FI
147660           CALL HXST                                  %*81F*
147661           A:=X+CHEAD=:D;=MASTB                       %BUFFER ADDRESS      *81F*
147665    HSPEC: X=:T; AD=:LISTF.LMEM                       % *H*
147670           A:=X-LISTP SHZ -2:=:X                      % CALCULATE INDEX IN DCB TABLE *H*
147674           * AAX DCBX; STT 0,X,B                      % SAVE DCB ADDR. IN DCB TABLE *H*
147676           X:=A                                       % X POINTS TO DMA LIST ENTRY *H*
147677           ERB=:X.LKEY                                % *81F*
147701           LISTF+4=:LISTF                             %LIST FILLER OF NEXT MESSAGE
147704           IF LISTF.LKEY=NLP THEN
147711                LISTP=:LISTF                          %REACHED END OF LIST
147713           FI
147713           IF ACTSW = 1 THEN GO FAR HDSTA FI          % TRANSMITTER ACTIVE
147720    HDENA: *TRR 10                                    % CLEAR CASCH
147721           IF LIINT.LKEY/\"3777"><"ERB" THEN
147727                GO FAR HDSTA                          % DO NOT START EMPTY LIST
147730           FI
147730    STARC: IF ACTSW = 0 THEN
147732                HXDOK/\MAINT; T:=HDEV+WRTC; *EXR ST   % CLEAR OLD GARBAGE
147737                LIINT; T:=HDEV+WDMA; *EXR ST          % START RECEIVER
147743                A:=1000; T+"WDCR-WDMA"; *EXR ST
147746                T+"RDCR-WDCR"; X:=-10; *EXR ST
147751                CALL LTOUT; *JAF *-2
147753                1=:ACTSW
147755           FI
147755    OUT1:  A:="1734"\/MAINT/\HXDOK
147760           T:=HDEV+WRTC; * EXR ST
147763           GO FAR HDSTA                               %ANY MORE MESSAGES ?
147764           *)FILL
147775    %
147775    %
147775    %       INPUT INTERRUPT DETECTED:
147775    %             GET STATUS
147775    %             LOG STATUS, USER DEVICE NUMBER AND 1.WORD IN FRAME
147775    %             SEND MESSAGE BACK TO USER.
147775    %
147775    %
147775
147775           DISP 0; INTEGER AC; PSID
147775           SYMBOL BUFSIZ=20
147775           SYMBOL TELMA=BUFSIZ-1
```

```
147775          INTEGER ARRAY BUFF0(BUFSIZ)                 % 1.WORD IN FRAME
150015          INTEGER ARRAY BUFF1(BUFSIZ)                 % DEVICE NUMBER
150035          INTEGER ARRAY BUFF2(BUFSIZ)                 % DEVICE STATUS
150055          INTEGER ARRAY BUFF3(11)                     % LIST KEYS WHEN DEVICE STOPPED
150066          INTEGER T1:=0
150067          INTEGER T2:=0
150070          INTEGER TELL:=0
150071          INTEGER T9:=0                               % DUMMY INTERRUPTS
150072          INTEGER POINTER ARRAY POBUFF:=BUFF0
150073  HIINT:  T:=HDEV+RRTS; *EXR ST                       % READ RECEIVER STATUS
150076          A=:HASTAT
150077          IF T:=ACTSW = 0 THEN MIN T9; P+0; GO OUT1 FI % EXPECTING NOTHING *81F*
150105          IF A/\ HX21M >< 0 THEN                      % X21-ERROR?
150107             *TRR 10                                  %*80B*
150110             A\/ LIINT.LKEY=:X.LKEY                   % YES, SAVE IT *H*
150113             IF A BIT HX21S THEN                      % X21 CLEAR INDICATION?
150115                HASTAT BONE BLDON=:HASTAT             % YES, SET BLOCK DONE TO
150120                LIINT.LKEY BONE XBLDN=:X.LKEY         % TO TERMINATE
150124             FI
150124          FI
150124          IF HASTAT/\"EMTY" >< 0 THEN
150127                0=:ACTSW                              % DEVICE STOPPED
150130                MIN STPCNT                            % LACK OF BUFFER, INC COUNTER
150131                P+0                                   % IN CASE OF SKIP
150132                LISTP=:T1; 0=:T2
150135                DO WHILE T1.LKEY >< "NLP"             % MAKE COPY OF DMA LIST
150142                   A=:BUFF3(T2)                       % WHEN DEVICE STOPPED
150144                   MIN T2; 0/\0; T1+4=:T1
150151                OD
150152          FI
150152  %
150152  %       LOOK AT RECEIVER MESSAGE
150152  %
150152  MORE:   *TRR 10                                     % CLEAR CACH
150153          A:=LIINT.LKEY=:D
150156          IF A NBIT XBLDN THEN                         % ANY MORE FILLED BLOCKS ?
150160                IF A = "ERB" THEN GO FAR STARC FI      % NO, ENABLE RECEIVER
150164                GO FAR OUT1
150165          FI
150165          MIN TELL; P+0
150167          TELL /\ TELMA=:TELL
150172          HDEV=:BUFF1(TELL)                           % LOG USER DEVICE NUMBER
150175          HASTAT=:BUFF2(TELL)                         % LOG DEVICE STATUS
150200          LIINT.LMEM2.AC=:POBUFF(TELL)                % LOG 1.WORD IN FRAME
150205          A:=LIINT.LBYTC+DISP1=:T                     % RECEIVED MESSAGE SIZE
150211          CALL XMPAT                                  % *H* GET DCB FROM DCB LIST
150212          X=:D; A:=LIINT.LKEY; 0=:X.LKEY; 0=:X.LMEM2;  X:=D
150220          IF A /\ "LMASK" = 3 THEN
150224                A:=0; CALL SCRET; CALL SADTS
150227          ELSE
150230                IF A BIT HX21S THEN EX21 ELSE EINP FI
150235                CALL SCRET
150236                D=:A; CALL SADTS; A\/DSTAT=:DSTAT
150242                HDERC+1=:HDERC
150245          FI
150245          X-BHEAD; CALL OCHAIN                                  % SEN MESSAGE BACK TO USER
150247          LIINT+4=:LIINT
150252          IF LIINT.LKEY=NLP THEN
150257                LISTP=:LIINT
150261          FI
```

```
150261          GO MORE
150262          *)FILL
150301    %     ARE WE RUNNING ?
150301    HXST: IF ACTSW = 0 THEN
150303            LISTP=:LISTF=:LIINT
150306              ELSE
150307            IF LISTF=LIINT THEN A:=ETOMUCH; GO FAR BACKX FI
150315          FI
150315          EXIT
150316    %
150316    %     SPECIAL ENTRY FOR PRIVILEGED USERS (MESSID<0)
150316    %     FIRST TWO DATA WORDS OF DCB POINTS TO RECEIVER BUFFER.
150316
150316    XSHDR: CALL ZXCHK                                  % CHECK DCB IS FROM XMSG
150317          CALL HXST
150320          T:=MASTB; * XHBUF@3 LDDTX
150322          GO FAR HSPEC
150323          RBUS
```

```
150327
150327   %=====================================================================
150327   % 40.13      S M C L E A R
150327
150327   %
150327   %PURPOSE:     TOTAL CLEAR OF INTERFACE
150327   %
150327   %OPERATION:   PERFORMS A DEVICE CLEAR. UNTREATED MESSAGES
150327   %             ARE SENT BACKX TO USER.
150327   %MESSAGE DECRIPTION:
150327   %           - FUNCTION = MCLEAR
150327   %           - NOT APPLICABLE
150327   %
150327   %ENTRY:    B-REG DATAFIELDPOINTER
150327   SUBR SMCLEAR
150327   SMCLEAR: T:=HDEV+WTCR; A:=2; *EXR ST; IOF        % SEND EOM TO AVOID YELLOW LIGTH
150334           A:=100; T:=HDEV+WRTC; *EXR  ST           %*80B*
150340           A:=40; *EXR ST                           %*80B*
150342           T+"WDCR-WRTC";*EXR ST; ION
150345           X:=OMSG; T:=MASTB; * XCFUN@3 LDATX
150350           IF A = 0 THEN                            % SPECIAL CLEAR (SUPER) *H*
150351             0=: HINIF =: OQUEU =: IQUEU =: WAKEF   % CLEAR EVERYTHING
150355             1=:INTSTA; DFOPP =: B                  % TAKE OPPOSITE DATAFILED
150361             0=: HINIF =: OQUEU =: IQUEU =: WAKEF   % CLEAR EVERYTHING
150365             1=;INTSTA; DFOPP =: B
150371             GO FAR HDSTA                           % GIVE NO RESPONCE TO USER
150372           ELSE
150373             1=:INTSTA; CALL SRRES                    %RESET THIS SIDE
150376             DFOPP=:B; 1=:INTSTA; CALL SRRES          %RESET OTHER SIDE
150403             DFOPP=:B; X:=OMSG
150406             A:=0 CALL SADTS GO FAR BACKX
150411           FI
150411   RBUS
```

```
150414
150414   %=============================================================================
150414   % 40.14      H D S I N
150414   %
150414   %PURPOSE: INITIATE INTERFACE AND SET MODUS
150414   %
150414   %MESAGE DECRIPTION:
150414   %           - FUNCTION = INIT
150414   %           - STATUS   = RETURN STATUS
150414   %           - ADDSTA   = HARDWARE STATUS
150414   %           - MODUS    = 0 - FULL DUPLEX
150414   %                        1 - HALF DUPLEX
150414   %                        2 - MAINTENANCE MODUS
150414   %           - CFSIZE   = MAX FRAMESIZE
150414   %           - IRTRY    = # OF RETRIES WANTED IF ERROR. NOT APPLICABLE FOR INPUT
150414   %           - IDISP    = DISPLACEMENT IN # OF BYTES
150414   %
150414   %ENTRY:     B-REG DATAFIELDPOINTER
150414   %           X-REG MESSAGE POINTER
150414   %
150414   %EXIT:      AS ENTRY
150414   %
150414   SUBR HDSIN
150414   HDSIN: IF INTSTA><1 THEN A:=ENCLEAR; GO FAR BACKX FI
150422          T:=MASTB; * IMODU@3 LDATX
150424          IF A=:D >> MAMOD THEN A:=EPAR; GO FAR BACKX FI
150432          IF A:=D = HALF THEN T:=40 ELSE T:=0 FI
150441          T=:CMODI                                        %SET HALF OR FULL DUPLEX
150442          IF A = MAMOD THEN A:=140 ELSE A:=100 FI         %*80B*
150450          A=:MAINT
150451          T:=HDEV+WRTC; *EXR ST                           %SET POSSIBLE MAINTENANCE MODE
150454          T:=MASTB; * IFSIZ@3 LDATX
150456          A=:MAXR                                          %MAX RECEIVER FRAMESIZE
150457          IF A < 1 THEN A:=EILFZ; GO FAR BACKX FI          % TOO SMALL MESSAGE
150464          0=:CHECK
150465          T:=MASTB; * IRTRY@3 LDATX
150467          A-=:XRETRY                                       %NUMBER OF RETRYES WANTED
150471          X-BHEAD; CALL RACTB; X+BHEAD
150474          IF A>15 THEN
150477            T:=MASTB; * IDISP@3 LDATX
150501              ELSE
150502            A:=0
150503          FI
150503          A=:DISP1+MAXR=:MAXR
150506          A:=XINITA; T:=HDEV+WDMA; *IOF; EXR ST
150513          400; T+"WDCR-WDMA";*EXR ST
150516          T+"RDCR-WDCR"; X:=-10;*EXR ST
150521          CALL LTOUT; * JAF *-2
150523          *ION; TRR 10                                    %CLEAR CASCH
150525          A;=CHECK                                        %CHECKSUM FROM INTERFACE
150526          IF A><INTCH THEN
150531                A=:OMSG.ADSTA
150533                A:=EILLINT; GO FAR BACKX
150535          FI
150535          X:=DFOPP                                        %UPPDATE OPPOSITE DATAFIELD
150536          XRETRY=:X.XRETRY
150540          MAXR=:X.MAXR
150542          MAINT=:X.MAINT; DISP1=:X.DISP1
150546          2=:X.INTSTA
150550          2=:INTSTA; A:=0; X:=OMSG; CALL SADTS; GO FAR BACKX
```

```
150556   RBUS
150564
150564   %=============================================================================
150564   % 40.15      T R A S E T    R E C S E T
150564
150564   %PURPOSE:     RESET TRANSMITTER
150564   %CHANGES FOR H VERSION: THE DCB FOR XMSG MAY CONTAIN A POINTER TO A DMA LIST,
150564   %                       THUS THE DCB LIST MUST BE USED FOR TAPPING
150564   SUBR TRASET,RECSET
150564   TRASET:       A:=0
150565           T:=HDEV+WTTC; *EXR ST
150570           A:=L=:XREENT; 0=:WAKEF
150573           IF X:=DCBX >< 0 THEN                          % ANY CURRENT DCB ?
150575             0=:DCBX; ECLEAR; CALL SCRET                 % YES, SEND IT BACK
150600             A:=0; CALL SADTS; X-BHEAD; T:=6; CALL SBYTC; CALL OCHAIN
150606           FI
150606           0=:TMR
150607           A:=XREENT=:P
150611   %
150611   %       RECSET
150611   %
150611   %PURPOSE:     RESET RECEIVER
150611   %
150611   RECSET:       A:=MAINT\/100                           %*80B*
150613           T:=HDEV+WRTC; *EXR ST
150616           0=:TMR; GO SENDB
150620           GO SENDB
150621   %
150621   %       SENDB
150621
150621   %PURPOSE:     SEND BACK (TO USER) UNTREATED MESSAGES
150621   %
150621   SENDB: LISTP=:LIINT
150623           A:=L=:XREENT                                  % TEMP SAVE OF L-REG
150625           DO WHILE LIINT.LKEY >< "NLP"
150632                 IF A >< 0 THEN                          % *H*
150633                    IF X.LMEM2 >< 0 THEN                 % WE DO NOT TRUST THE KEY
150635                                  0=:X.LMEM2
150636                                  0=:X.LKEY=:X.LBYTC     % X POINTS TO DMA ENTRY
150640                                  T:=0; CALL XMPAT       % FIND DCB ENTRY AND SET LENGTH
150642                                  ECLEAR; CALL SCRET     % X POINTS TO DCB
150644                                  A:=0; CALL SADTS
150646                                  X-BHEAD; T:=6; CALL SBYTC
150651                                  CALL OCHAIN
150652                       FI
150652                 FI
150652                 LIINT+4=:LIINT
150655           OD
150656           A:=XREENT=:L
150660           A:=0=:ACTSW
150662           EXIT
150663   RBUS
```

```
150672
150672   %=====================================================================
150672   % 40.16      S P S T A
150672   %
150672   %PURPOSE:     GIVE PORT STATUS
150672   %
150672   %MESSAGE DECRIPTION:
150672   %        - FUNCTION   = PSTAT
150672   %        - STATUS     = NOT APPLICABLE
150672   %        - ADDSTA     = NOT APPLICABLE
150672   %        - ERRNO      = TOTALE # OF ERRORS (RELATED TO ADSTA)
150672   %        - ORERR      = OR FUNCTION OF ALL ERRORS (RELATED TO ADSTA)
150672   %        - LHAST      = LAST HARDWARE STATUS (RRTS OR RTTS)
150672   %        - SSTPC      = RECEIVER STOPP COUNTER
150672   %        - MAXEM      = MAX NUMBER OF EMTY BUFFERS HELD BY RECEIVER. (LIST LENGTH
150672   %        - HHMAX      = BUFFER SIZE IN BYTES FOR THIS LDN
150672   %        - HHEAD      = SYSTEM PART (REDUNDENCE) OF EACH DCB
150672   %
150672   %
150672   SUBR SPSTA
150672   SPSTA: X-BHEAD; CALL RBYTC; X+BHEAD                  % GET BYTECOUNT
150675          IF A<20 THEN
150700                ETOSM; CALL SCRET                           % MESSAGE TOO SMALL
150702                A:=0; CALL SADTS; GO SP1
150705          FI
150705          X+CHEAD =: OMSG                               % POINT TO USER INFORMATION
150707          IF A >=24 THEN
150712                T:=MASTB; MAX; * HHMAX@3 STATX          % IN BYTES
150715                "BHEAD"+2 SH 1; * HHEAD@3 STATX         % IN BYTES
150721          FI
150721          T:=MASTB
150722          A:=HDERC; * ERRNO@3 STATX
150724          A:=DSTAT; * ORERR@3 STATX
150726          A:=HASTAT; * LHAST@3 STATX
150730          A:=STPCNT; * SSTPC@3 STATX
150732          A:=LISTL; * MAXEM@3 STATX
150734          0=:HDERC=:DSTAT=:STPCNT
150737          X-CHEAD; A:=0; CALL SCRET; CALL SADTS
150743          X-BHEAD; T:=24; CALL SBYTC
150746   SP1:   CALL OCHAIN; GO HDSTA
150750   RBUS
150756
150756   *"8PACL
150756    %=====================================================================
150756   %            3 S E T C L O C K
150756   %
150756   % SUBROUTINE TO THE SETCLOCK ROUTINE
150756   % (PLACED IN POF-AREA TO SAVE SPACE IN RESIDENT)
150756   %
150756   SUBR 3SETCLOCK
150756   INTEGER POINTER LREG
150757   3SETCLOCK:
150757          A:=L=:"LREG"
150761          IF CPSTA BIT 5N100 THEN
150764             A:=0; CALL PANEL
150766             IF A BIT 17 THEN          % PANEL PRESENT
150770                X:=-1; ACL7(X)-TBASE(X)=:ADDYEAR; 0=:DAYES
150775                DO WHILE ADDYEAR>0
151000                   0=:PERIOD
151001                   FOR PERIOD TO 3 DO WHILE ADDYEAR>0
```

```
151010                      DAYEAR(PERIOD)+DAYES=:DAYES
151014                      ADDYEAR-1=:ADDYEAR
151017                   OD
151023                OD
151024                IF PERIOD=1 THEN 35 ELSE 34 FI A=:DAMONTH(1)
151035                X:=-2; ACL7(X)-TBASE(X)-1=:ADDMONTH
151042                FOR X:=0 TO ADDMONTH DO
151046                   DAMONTH(X)+DAYES=:DAYES
151051                OD
151053                X:=-3; ACL7(X)-TBASE(X)+DAYES=:DAYES SHZ 1=:HDAYES
151062                X:=-4; A:=ACL7(X)=:HOURS=:D:=0; T:=14; *RDIV ST
151071                A+HDAYES=:HDAYES; A:=D=:HOURS*7020=:SECOND; X:=-5
151100                ACL7(X)*74+SECOND=:SECOND; X:=-6; ACL7(X)+SECOND=:SECOND
151110                GO PANUP; *)FILL
151127
151127   PANUP:       SECOND; A=:D/\377\/2000; CALL PANEL
151134                A:=D; A SHZ -10\/2400; CALL PANEL
151140                HDAYES; A=:D/\377\/3000; CALL PANEL
151145                A:=D; A SHZ -10\/3400; CALL PANEL
151151             FI
151151          FI; GO LREG
151152   RBUS
151162
151162
151162   %-============================================================================
151162   %             3 R C L O C K
151162   %
151162   % SUBROUTINE TO THE READCLOCK ROUTINE IN RESIDENT
151162   %
151162   SUBR 3RCLOCK
151162   INTEGER POINTER LREG=?
151162   3RCLOCK: A:=L=:"LREG"
151164           A:=0; CALL PANEL
151166           IF A NBIT 17 THEN GO LREG FI                  % PANEL NOT PRESENT
151171           A:="22000"; CALL PANEL                        % READ LOW SECONDS
151173           A/\377=:D; A:="22400"; CALL PANEL             % READ HIGH SECONDS
151177           A SHZ 10\/D=:SECOND; A:="23000"; CALL PANEL% READ LOW HDAYS
151204           A/\377=:D; A:="23400"; CALL PANEL             % READ HIGH HDAYS
151210           A SHZ 10\/D=:HDAYES
151213           "PX9CL"; CALL EX113M                          % READ SINTRAN CLOCK
151215           0=:DAYES; GO L1; *)FILL
151232
151232   INTEGER POINTER LREG
151233
151233   %  UPDATE CALENDER
151233
151233   L1:     X:=-1; PXCLX(X)-TBASE(X)=:ADDYEAR
151237           DO WHILE ADDYEAR>0
151242              0=:PERIOD
151243              FOR PERIOD TO 3 DO WHILE ADDYEAR>0
151252                 DAYEAR(PERIOD)+DAYES=:DAYES
151256                 ADDYEAR-1=:ADDYEAR
151261              OD
151265           OD
151266           IF PERIOD=1 THEN 35 ELSE 34 FI; A=:DAMONTH(1)
151277           X:=-2; PXCLX(X)-TBASE(X)-1=:ADDMONTH
151304           FOR X:=0 TO ADDMONTH DO
151310              DAMONTH(X)+DAYES=:DAYES
151313           OD
151315           X:=-3; PXCLX(X)-TBASE(X)+DAYES=:DAYES SHZ 1=:T
```

```
151324          IF HDAYES-T=:HDAY2 < 0 THEN                 % PANEL CLOCK SHOULD
151330  ERR:       ER205=:ERUCL                             % BE GREATER THAN
151332             GO OUT; *)FILL                           % SINTRAN CLOCK
151346          FI
151346          X:=-4; PXCLX(X)=:D:=0; T:=14; *RDIV ST
151354          T:=HDAY2-A=:HDAY2; A:=D*7020=:SEC2
151362          X:=-5; PXCLX(X)*74+SEC2=:SEC2
151367          X:=-6; PXCLX(X)+SEC2=:T; GO LABL1; *)FILL
151401
151401  OUT:    GO LREG
151402
151402  LABL1:  IF SECOND>>=T THEN
151405             A-T
151406          ELSE
151407             X:=HDAY2-1=:HDAY2; X:=124300-T; A+X
151415          FI; D:=0; AD SHZ -1; A=:XCLNUN
151420          IF HDAY2<0 GO ERR
151422          2=:XCLUN; "PLXCL"; CALL EX112M
151426          IF D><0 THEN XCLNUN+1=:XCLNUN FI; "PLXCL"; CALL EX112M
151435          T:=14; A:=HDAY2; *RMPY SA DT
151440          AD=:DXCLUN; 4=:XCLUN; T:=30=:XCLNUN
151445          DO
151445             AD:=DXCLUN
151446          WHILE A><0 OR D>>30
151452             D-T; A:=A+C-1; AD=:DXCLUN; "PLXCL"; CALL EX112M
151460          OD; A;=D=:XCLNUN; "PLXCL"; CALL EX112M
151465          GO OUT
151466  RBUS
151476
151476  *"
151476
151476  *"8GPI0+8GPI1+8GPI2+8GPI3+8GPI4+8GPI5+8GPI6+8GPI7 8XMSG
151476   SUBR GPIBT,GPIBH,GPBUS,GPFIN,GPERR
151476  %=================================================================
151476  %                                                                =
151476  % EXTENDED TRANSFER ROUTINE FOR GENERAL PURPOUSE INTERFACE BUS    =
151476  %                                                                =
151476  %=================================================================
151476  % FUNCTION CODES:
151476  %
151476  %                        00 : DMA INPUT
151476  %                        01 : DMA OUTPUT
151476  %                        02 : READ STATUS
151476  %                        03 : CLEAR DEVICE
151476  %                        04 : READ PIO
151476  %                        05 : WRITE PIO
151476  %                        06 : ENABLE SRQ INTERRUPT
151476  %                        07 : DISABLE ALL INTERRUPTS
151476  %                        10 : SERIAL POLL
151476  %                        11 : PARALLELL POLL
151476  %                        12 : READ SYSTEM DEVICE LIST
151476  %                        13 : READ USER DEVICE LIST
151476  %                        14 : SEND CONTROL STRING WITHOUT DEVICE LIST
151476  %                        15 : SEND COMMAND BYTE WITH DEVICE LIST
151476  %                        16 : RUN MICROPROGRAM CONTROLLER TEST
151476  %                        17 : LOG OFF
151476  %                        20 : LOG ON AS PRIVILEGED
151476  %                        21 : LOG ON AS UNPRIVILEGED
151476  %                        22 : WAIT FOR SRQ INTERRUPT (CODE NEVER SEEN BY TRANFER ROUTINE)
151476  %                        23 : CHECK FOR SRQ INTERRUPT (CODE NEVER SEEN BY TRANFER ROUTINE)
```

```
151476    %                       24 : SET SINGLE USER MODE
151476    %                       25 : RESET SINGLE USER MODE
151476    %                       26 : WHO IS ON
151476    %=============================================================
151476
151476
151476    %  GLOBAL DISP VARIABLES ARE DECLARED AS NEGATIVE
151476    DISP 12
151476      INTEGER GPIBNO                % CONTROLLER NO (ASCII)
151476      INTEGER ACTIV                 % ACTIV FLAG
151476      INTEGER XTBLK                 % XT-BLOCK ADDRESS
151476      INTEGER SXTBL                 % SRQ-TASK XT-BLOCK ADDRESS
151476      INTEGER CPORT                 % COMMAND-PORTNUMBER
151476      INTEGER SQTXA                 % SRQ-TASK XT-BLOCK ADDRESS
151476      INTEGER DPORT                 % DMA-PORTNUMBER
151476      INTEGER DMESA                 % DMA MESSAGE ADDRESS
151476      DOUBLE  DBUFA                 % DMA BUFFER ADDRESS
151476      INTEGER MVERS                 % MICROPROGRAM VERSION AND REV. LEVEL
151476      INTEGER SUSFL                 % SINGLE USER FLAG
151476      INTEGER DEVFL                 % DEVICE FLAG
151476      INTEGER USCONT                % NUBER OF USERS PRESENTLY LOGGED ON
151476      INTEGER CGPIM                 % CURRENT MESSAGE ADDRESS
151476      INTEGER CURMTY                % CURRENT MESSAGE TYPE
151476      DOUBLE  CURMN                 % CURRENT MAGIC NUMBER
151476      INTEGER ERADD                 % LAST ERROR ADDRESS
151476      INTEGER XERCO                 % X-MESSAGE ERRORCODE
151476      INTEGER SRQFL                 % SRQ PROCESSING FLAG
151476      INTEGER POINTER MRETA         % RETURN ADDRESS AFTER MONITOR CALL
151476      INTEGER POINTER SRETA         % RETURN ADDRESS AFTER MONITOR CALL
151476      INTEGER POINTER GRETA         % RETURN ADDRESS AFTER TIMEOUT ROUTINE
151476      INTEGER 9TREG                 % SAVED TREG WHEN CALLING ID11/WT11
151476      INTEGER 9AREG                 % SAVED AREG WHEN CALLING ID11/WT11
151476      INTEGER 9DREG                 % SAVED DREG WHEN CALLING ID11/WT11
151476      INTEGER 9XREG                 % SAVED XREG WHEN CALLING ID11/WT11
151476      INTEGER DINPT                 % INPUT DATA TRANSFER
151476      INTEGER PIODP                 % PIO DATA POINTER
151476      INTEGER PIOWC                 % PIO WORD COUNTER
151476      INTEGER FIRST                 % FIRST WORD TO BE RECEIVED FLAG
151476      INTEGER POINTER SRSTR         % START LOCATION FOR SRQ ROUTINE
151476      INTEGER ARRAY UMESS(20)       % USER MESSAGE TABLE
151476      DOUBLE  ARRAY MNARR(20)       % MAGIC NUMBER ARRAY
151476        INTEGER CURFU               % CURENT FUNCTION CODE
151476        INTEGER CGPUS               % CURENT USER NO.
151476        INTEGER ARRAY INSTA(10)     % INSTRUMENT NO. ARRAY
151476        DOUBLE  CURAD               % CURENT DATA ADDRESS
151476        INTEGER CURBC               % NUMBER OF BYTES TO BE TRANSFERED
151476        DOUBLE  ARRAY UENV(20)      % USER TERMNO / RT ADDRESS TABLE
151476        DOUBLE  DINSO=CURMN         % PHYS. MEMORY ADDRESS FOR RETURN DATA ARRAY
151476        INTEGER GABSF=USCONT        % FLAG SET IF DRIVER USED BY ABSTR
151476    PSID
151476    INTEGER SVERS:=102                      % MICROPROGRAM VERSION THAT WILL WORK WITH THIS SOFTWARE
151477    INTEGER MSLOF
151500    INTEGER XRMES:=3                        % FIRST TWO BYTES IN XROUT MESSAGE
151501    INTEGER XRLNG:=10                       % NEXT TWO BYTES IN XROUT MESSAGE
151502    INTEGER XRP01:=177406                   % PARAMETER LENGTH IN XROUT MESSAGE
151503    INTEGER ARRAY NAME:='GPIBXX'            % PORT NAME
151507
151507    SYMBOL MUSER=1
151507    SYMBOL MINNO=2
151507    SYMBOL MADDR=12
```

```
151507    SYMBOL MNOBY=14
151507    SYMBOL MDATA=15
151507
151507
151507
151507
151507    % INSTRUCTION DEFINITIONS
151507
151507    SYMBOL GPLON=101              % LOG ON USER
151507    SYMBOL GPLOF=105              % LOG OFF USER
151507
151507    %=============================================================================
151507    %==================== I N I T   G P I B   C O N T R O L L E R ==============
151507    %=============================================================================
151507
151507    GPIBT: A:=XTBLK;IF A><0 GO WT11                                  % IF GPIB ALLREADY STARTED GO WT11
151512           0=:CGPIM=:CGPUS=:SUSFL=:GPRUN=:USCONT                     % RESET CURRENT MESSAGE POINTER, USER
151517           "SRENT"=:"SRSTR"                                          % INITIATE POINTER TO SRQ INTERUPT RO
151521           T:=3;CALL TRNSF;CALL FATER;GO WAIT1;GO RMTST               % CLEAR GPIB CONTROLLER
151526    WAIT1: CALL GTSUB;IF T+1=0 THEN T:=105;CALL FATER FI             % WAIT FOR STATUS
151534           GO GPIBT
151535    RMTST: T:=16;CALL TRNSF;CALL FATER;GO WAIT2;GO ILOF               % RUN MICROPROGRAM TEST
151542    WAIT2: CALL GTSUB;IF T+1=0 THEN T:=105;CALL FATER FI             % WAIT FOR STATUS
151550           GO RMTST
151551    ILOF:  T=:MVERS;T SHZ -10=:A;A-SVERS
151555           IF A><0 THEN
151556              A:=-1=:SXTBL;A:=T;A SHZ 10 \/ SVERS =:MVERS;GO INXM     % IF WRONG MICROPROGRAM SET SXTBL TO
151565           FI
151565           A:="MSLOF"=:D;A:=0;AD=:CURAD
151571           A:=GPLOF SHZ 10=:MSLOF;A:=1=:CURBC
151576    ILOF1: T:=14;CALL TRNSF;CALL FATER;GO WT22;GO INXM
151603    WT22:  CALL GTSUB;IF T+1=0 THEN T:=105;CALL FATER FI
151611           GO ILOF1
151612
151612    *)FILL
151620
151620    %=============================================================================
151620    %================ S E T   U P   X M S G   C O M U N C A T I O N ==============
151620    %=============================================================================
151620
151620    INXM:  L:=0;T:=XFOPN;*MON 2XMSG;JPL I (WT11                      % OPEN COMMAND PORT
151624           IF T<0 THEN CALL XERR FI                                  % CHECK IF ERROR RETURN
151627           A=:CPORT;A:=L=:XTBLK                                      % SAVE PORT NO. AND XTBLK ADDRESS
151632           T:=XFOPN;CALL EGXMS;A=:DPORT                              % OPEN DMA  PORT
151635           T:=XFGET;A:=GPDZI;A+6;CALL EGXMS                          % GET DMA BUFFER
151641           A=:DMESA=:X CALL GDRXM(XDINF);CALL XSERR                  % GET DMA BUFFER ADDRESS
151646           AD=:DBUFA                                                 % SAVE DMA BUFFER ADDRESS
151647           X:=2;A:=GPIBNO;A=:NAME(X)                                 % WRITE GPIBNO. IN NAME
151652           A:=14=:D;A:="XRMES";X:=0;T:=XFWRI;CALL EGXMS              % WRITE TO MESSAGE
151660           T:=XFSND BONE XFROU;X:=CPORT;CALL EGXMS                   % NAME PORT
151664           T:=XFRCV BONE XFWTF;A:=CPORT;CALL EGXMS                   % REC. RESPONS FROM XROUT
151670           AD:=DBUFA;*EXAM                                           % READ XROUT STATUS
151672           IF T><0 THEN  CALL XRERR FI                               % IF T><0 FATAL ERROR
151675           A:=SXTBL;IF A+1=0 GO MVERR                                % IF MICROPROGRAM VERSION ERROR DONT
151700           L:=0;T:=XFOPN;*MON 2XMSG;JPL I (WT11                      % OPEN SRQ CPORT
151704           IF T<0 THEN CALL XERR FI                                  % CHECK IF ERROR RETURN
151707           A=:SQTXA;A:=L=:SXTBL                                      % SAVE PORT NO. AND XTBLK ADDRESS
151712           T:=XFGET;A:=6;CALL SCALL                                  % GET SRQ MESSAGE
151715    MVERR: FOR X:=0 TO 17 DO
151721              0=:UMESS(X)                                            % ZERO USERTABLE ENTRY
```

```
151722          OD
151724          A:=0=:D
151726          FOR X:=0 STEP 2 TO 36 DO
151732             AD=:UENV(X)
151733          OD
151735          0=:SRQFL
151736          -1=:GPRUN
151740          GO GGPIC
151741
151741    *)FILL
151752
151752    %=========================================================================
151752    %============== W A I T   F O R   C O M M A N D ==========================
151752    %=========================================================================
151752
151752
151752    GGPIC:   T:=SXTBL;IF T+1=0 GO RGPIC                                  % IF MICROPROGRAM VERSION ERROR DONT
151756           1:=6;CALL TRNSF;CALL FATER;CALL FATER                         % ENABLE SRQ INTERRUPT IF SRQFL >< 0
151762    RGPIC:   0=:CGPIM;T:=XFRCV BONE XFWTF;A:=CPORT;CALL EGXMS             % WAIT FOR COMMAND
151767           T=:CURMTY;A:=D=:CGPIM                                         % SAVE CURRENT MESSAGE ADDRESS AND TY
151772           A:=SRQFL;IF A><0 THEN CALL WT11 FI                            % CHECK IF SRQINTERRUPT IS BEING PROC
151775
151775    %-------------------------------------------------------------------------
151775    %============== G E T     C O M M A N D ==================================
151775    %-------------------------------------------------------------------------
151775
151775    PRCOM:  T:=7;CALL TRNSF;CALL FATER;CALL FATER                         % DISABLE SRQ INTERRUPT IF SRQFL >< 0
152001           A:=CGPIM;
152002           T:=XFMST;CALL EGXMS                                           % GET MESSAGE STATUS
152004           AD=:CURMN                                                     % SAVE MAGIC NUMBER
152005           A:=32=:D;A:=B+"CURFU";X:=0;T:=XFREA;CALL EGXMS                % GET PARAMETER BLOCK INTO DATAFIELD
152014           X:=CGPIM;CALL GDRXM(XDINF);CALL XSERR                         % GET BUFFER ADDRESS
152020           X:=MDATA;*RADD SX DD;COPY SA DA ADC;STD CURAD,B               % SAVE BUFFER ADDRESS
152024           0=:DINPT;A:=CURMTY
152026
152026    %=========================================================================
152026    %============== D E C O D E   C O M M A N D ==============================
152026    %=========================================================================
152026
152026
152026           IF A=XMTRE THEN 17=:CURFU;GO FAR LOGOF FI                     % IF RETURNED MESSAGE GO LOGOF
152034           IF A=XMROU GO FAR LOGON                                       % IF ROUTED MESSAGE GO LOGON
152037           T:=SXTBL;IF T+1 = 0 THEN                                      % CHECK IF MICROPROGRAM VERSION ERROR
152043           T:=110;GO FAR ERRET FI                                        % RETURN ERROR CODE
152045           A:=CURFU;IF A=3 GO FAR CLDEV                                  % CHECK IF CLEAR DEVICE
152051           IF A<=1 GO FAR DMAT                                           % CHECK IF DMA TRANSFER
152054           IF A=16 GO FAR MICTE                                          % CHECK IF RUN MICRO TEST
152057           IF A=17 GO FAR LOGOF                                          % CHECK IF LOGOFF
152062           IF A=24 GO FAR SSUSE                                          % CHECK IF SET SINGLE USER
152065           IF A=25 GO FAR RSUSE                                          % CHECK IF RESET SINGLE USER
152070           IF A=26 GO FAR RETUR                                          % CHECK IF WHO-IS-ON COMMAND
152073           IF A=27 GO FAR TRANS                                          % CHECK IF TRANSF BETWEEN DEVICES
152076           T:=CURFU;GO GDRIV
152100
152100    *)FILL
152120
152120    %=========================================================================
152120    %============== C A L L   D R I V E R ====================================
152120    %=========================================================================
152120
```

```
152120    GDRIV: CALL TRNSF;GO ERROR;GO BUSY;GO FINISH
152124    GPBUS: CALL GTSUB                                          % CALL DRIVER
152125           IF T+1=0 THEN T:=105;0=:CURBC;GO ERRET FI
152133           T:=9TREG;GO GDRIV
152135    GPFIN: A:=CURFU;IF A-11=0 THEN  T=:INSTA(0) FI
152142           T:=0
152143
152143
152143    %=============================================================================
152143    %---------------- R E T U R N   S T A T U S   T O   U S E R ------------------
152143    %=============================================================================
152143
152143    RETUR: A:=CURFU;IF A-17=0 GO GGPIC
152146           IF A-1=0 OR A-1=0 THEN                                % IF LOGOF WAIT FOR NEXT COMMAND
152152              X:=CGPUS;A:=CGPIM=:UMESS(X)                        % IF LOGON STORE MESSAGEADDRESS IN US
152155           FI
152155           A:=CURFU;X:=DINPT;IF A=0 AND X BIT 17 THEN
152162              T:=XFSCM;A:=DPORT=:D;A:=DMESA;CALL EGXMS            % CHECK IF DMA INPUT AND BYTCONT > GP
152167              T:=XFWHD;A:=CURBC=:D;A:=0;X:=0;CALL EGXMS           % SET DMA BUFFER AS CURRENT MESSAGE
152175              X:=DMESA;A:=-1;CALL GDRXM(XDSBP);CALL XSERR         % WRITE STATUS IN DMABUFFER
152202              X:=DMESA;A:=CURBC+6;CALL GDRXM(XDSBP);CALL XSERR    % SET MESSAGE SIZE TO ZERO
152210              T:=XFSND BONE XFSEC;X:=DPORT;AD:=CURMN;CALL EGXMS   % SET CORRECT SIZE OF BUFFER
152215              T:=XFRCV BONE XFWTF;A:=DPORT;CALL EGXMS             % SEND DMA BUFFER TO USER
152221              A:=T;IF A=XMTRE THEN 17=:CURFU;GO FAR LOGOF FI      % WAIT FOR DMA BUFFER
152230              T:=XFSCM;A:=CPORT=:D;A:=CGPIM;CALL EGXMS            % IF RETURNED MESSAGE GO LOGOF
152235              0=:DINPT;T:=0                                       % SET USERS MESSAGE AS CURRENT MESSAG
152237           FI
152237
152237    %=============================================================================
152237    %============== R E T U R N   S T A T U S   T O   U S E R ================
152237    %=============================================================================
152237
152237
152237    ERRET: A:=CURFU;IF A-17=0 THEN
152242              CALL NFERR;GO FAR GGPIC                            % IF LOGOF WRITE ERROR MESS AND WAIT
152244           FI                                                    % WRITE ERROR MESSAGE
152244           A:=CURFU;X:=32; IF A-26=0 THEN
152250           X:=20=:CURBC;X:=72                                    % IF WHO IS ON COMMAND, ALSO SEND TER
152253           FI
152253           T=:CURFU;D:=X;A:=B+"CURFU";X:=0;T:=XFWRI;CALL EGXMS
152262           T:=DINPT;IF T><0 THEN                                 % WRITE STATUS IN MESSAGE
152265              X:=CGPIM;A:=-1;CALL GDRXM(XDSBP);CALL XSERR
152272              X:=CGPIM;A:=CURBC+32;CALL GDRXM(XDSBP);CALL XSERR  % SET MESSAGE SIZE TO ZERO
152300           FI                                                    % SET CORRECT SIZE OF BUFFER
152300           T:=XFSND BONE XFSEC;X:=CPORT;AD:=CURMN;CALL EGXMS
152305           GO FAR GGPIC                                          % RETURN MESSAGE
152306
152306    *)FILL
152316
152316    %=============================================================================
152316    %---------------- R U N   M I C R O P R O G R M   T E S T ------------------
152316    %=============================================================================
152316
152316    MICTE: A:=CGPUS;IF A><0 THEN T:=102;GO ERRET FI
152322           A:=USCONT;IF A > 1 THEN T:=112;GO ERRET FI            % CHECK IF PRIVILIDGED
152330           T:=CURFU;GO FAR GDRIV
152332
152332    %=============================================================================
152332    %======= T R A N S F E R   D A T A   B E T W E E N   D E V I C E S =======
152332    %=============================================================================
```

```
152332
152332  TRANS: AD:=DBUFA;D:=:A;A+3;*COPY SD DD ADC;SWAP SA DD;STD CURAD,B     % STORE DMA BUFFER ADDRESS IN DATAFIE
152340         T:=0;GO FAR GDRIV
152342
152342  %=============================================================================
152342  %============== C L E A R   C O N T R O L L E R ========================
152342  %-----------------------------------------------------------------------------
152342
152342  CLDEV: A:=CGPUS;IF A><0 THEN T:=102;GO ERRET FI                          % CHECK IF PRIVILIDGED
152346         T:=CURFU;GO FAR GDRIV
152350
152350  %=============================================================================
152350  %============== S E T   S I N G L E   U S E R   M O D E ==================
152350  %=============================================================================
152350
152350  SSUSE: A:=USCONT; IF A>1 THEN T:=112;GO ERRET FI;                        % CHECK IF OTHER USERS IS ON
152356         T:=1=:SUSFL:=0; GO FAR RETUR
152362
152362  %-----------------------------------------------------------------------------
152362  %============== R E S E T   S I N G L E   U S E R   M O D E ================
152362  %-----------------------------------------------------------------------------
152362
152362  RSUSE: 0=:SUSFL=:T; GO FAR RETUR
152365
152365  %=============================================================================
152365  %============== D M A   T R A N S F E R ====================================
152365  %=============================================================================
152365
152365  DMAT:  A:=CURBC;T:=GPUZI;IF A<=T THEN T:=CURFU;GO FAR GDRIV FI        % CHECK IF BYTECOUNT > GPUZI
152373         A:=CURBC;T:=GPDZI;IF A>T THEN T:=107;GO ERRET FI              % CHECK IF BYTECOUNT > GPDZI
152401         AD:=DBUFA;D:=:A;A+3;*COPY SD DD ADC;SWAP SA DD;STD CURAD,B     % STORE DMA BUFFER ADDRESS IN DATAFIE
152407         A:=CURFU;IF A=1 GO UTDMA                                      % CHECK IF OUTPUT
152413         A:=DINPT BONE 17 =:DINPT;T:=CURFU;GO FAR GDRIV                % SET BIT 15 IN DINPT TO FLAG DMA IN
152420  UTDMA: T:=XFSCM;A:=DPORT=:D;A:=DMESA;CALL EGXMS                       % SET DMA BUFFER AS CURRENT MESSAGE
152425         T:=XFSND BONE XFSEC;AD:=CURMN;X:=DPORT;CALL EGXMS              % SEND DMA BUFFER TO USER
152432         T:=XFRCV BONE XFWTF;A:=DPORT;CALL EGXMS                        % WAIT FOR DMA BUFFER
152436         A:=T;IF A=XMTRE THEN 17=:CURFU;GO FAR LOGOF FI                 % IF RETURNED MESSAGE GO LOGOF
152445         T:=XFSCM;A:=CPORT=:D;A:=CGPIM;CALL EGXMS                       % SET COMMAND BUFFER AS CURRENT MESSA
152452         T:=CURFU;GO FAR GDRIV
152454
152454
152454  INTEGER SCRATCH              % SCRATCH LOCATION FOR FUNCTION 14 INITIATED BY TRANSFER ROUTINE
152455
152455  *)FILL
152461
152461  %=============================================================================
152461  %========== E R R O R   R E T U R N   F R O M   D R I V E R ================
152461  %=============================================================================
152461
152461  GPERR: 0=:TMR
152462         A:=CURMTY;IF A-XMROU = 0 THEN
152465            CALL TREMO                                                  % IF ERROR ON LOG ON, CLEAR TABLES
152466         FI
152466         GO FAR ERRET
152467  %=============================================================================
152467  %========== L O G   O N   N E W   U S E R ==================================
152467  %-----------------------------------------------------------------------------
152467
152467  LOGON: T:=SXTBL;IF T+1=0 THEN
152473         T:=110;GO FAR ERRET                                            % IF MICROPROGRAM VERSION ERROR RETUR
```

```
52475           FI
52475           A:=CURFU SHZ -10
52477           IF A=20 GO PRIUS
52502           IF A><21 THEN T:=100;GO FAR ERRET FI
52507           FOR X:=1 TO 17 DO
52513           A:=UMESS(X);IF A=0 GO UFOUN OD
152517    NOUSP: T:=103;GO FAR ERRET                                    % NO USER SPACE
152521    PRIUS: X:=0;A:=UMESS(X);IF A><0 THEN T:=104;GO FAR ERRET FI
152526    UFOUN: A:=SUSFL;IF A><0 THEN
152530             A:=USCONT; IF A><0 THEN
152532               T:=111;GO FAR ERRET
152534             FI
152534           FI
152534           A:=CGPIM;T:=XFREL;CALL EGXMS                          % RELEASE USER MESSAGE
152537           A:=GPUZI+32;T:=XFGET BONE XFWTF;CALL EGXMS             % GET NEW USER MESSAGE WITH CORRECT S
152544           A=:CGPIM=:UMESS(X)                                    % STORE MESSAGE IDENT IN USER ARRAY
152546           D:=A;A:=CPORT;A:=:D;T:=XFSCM;CALL EGXMS               % SET CURRENT MESSAGE
152553           X=:CGPUS
152554           T:=CGPUS SHZ 1;INSTA(4)=:D;INSTA(5);AD=:UENV(T)       % STORE TERMINAL NO. / RT DESC. ADDRE
152565           A:=GPDZI=:INSTA(0);A:=GPUZI=:INSTA(1)                 % STORE USERNO. AND BUFFER SIZES IN M
152573           A:=MVERS=:INSTA(2);A:=DEVFL=:INSTA(3)                 % STORE MICROPROGRAM VERSION AND DEVI
152601           A:=CGPUS SHZ 1=:X;AD:=CURMN=:MNARR(X)                 % STORE USERS MAGIC NUMBER
152606           A:=USCONT +1 =:USCONT
152611           A:=GPLON SHZ 10;A=:SCRATCH;A:="SCRATCH"=:D;A:=0
152617           AD=:CURAD;A:=1=:CURBC;T:=14;GO FAR GDRIV
152624
152624    %=============================================================================
152624    %========== L O G   O F F   U S E R ========================================
152624    %=============================================================================
152624
152624    LOGOF: A:=CGPIM=:D;FOR X:=0 TO 17 DO
152632           A:=UMESS(X);IF A=D GO ULOF OD
152637           A:=D;T:=XFREL;CALL EGXMS;GO FAR GGPIC
152643    ULOF:  X=:CGPUS;CALL TREMO;T:=XFREL;CALL EGXMS
152647           A:=GPLOF SHZ 10;A=:SCRATCH;A:="SCRATCH"=:D;A:=0;AD=:CURAD
152656           A:=1=:CURBC;T:=14;GO FAR GDRIV
152662
152662    *)FILL
152670    %=============================================================================
152670    %========== S R Q / D E V   I N T E R R U P T   E N T R Y ==================
152670    %=============================================================================
152670
152670    SRGET: T:=XFGET;A:=6;CALL SCALL                                % GET NEW MESSAGE
152673    SRENT: 0=:CGPUS=:CGPIM
152675    SRCDR: T:=10=:CURFU=:SRQFL
152700           CALL TRNSF;GO SRERR;GO WAIT4;GO PSREC
152704    WAIT4: CALL GTSUB;IF T+1=0 THEN T:=105;GO SRERR FI
152712           GO SRCDR
152713    PSREC: T=:CURFU;IF T+1=0 GO ENSIN                              % CHECK IF NON EKSPECTED SRQ-INTERRRU
152717           A:=CURFU;AD SHZ -6;A SHZ 1;A BONE "0";A BONE 11;AD SHZ 6  % UNPAC POLLSTATUS AND SET BIT 6 AND
152725           X:=A;A:=GPIBNO;AD SHZ -10;A -60 SHZ 3;A:=:D
152733           A SHZ -10; A -60;D+A;A:=X
152737           T:=XFWHD;X:=0;CALL SCALL
152742           A:=CURFU SHZ -14;A SHZ 1=:X;AD:=MNARR(X)               % GET USERS MAGIC NUMBER
152747           X:=SQTXA;T:=XFSND;CALL SCALL                           % SEND MESSAGE TO USER
152752           T:=XFGET;A:=6;CALL SCALL;T:="SRENT"=:"SRSTR"                                % GET N
152757    ENSIN: 0=:SRQFL;T:=CGPIM;IF T><0 GO FAR PRCOM                                % CHECK IF NEW COMMAND IS RE
152763           T:=6;CALL TRNSF;CALL FATER;CALL FATER;CALL ID11
152770           GO SRSTR
152771    SXERR: T=:;A:=300;T+A;A:="SRGET"=:"SRSTR"                      % XMSG ERROR
```

```
152776   SRERR: CALL NFERR;GO ENSIN                                              % WRITE ERRORMESSAGE
153000   %=======================================================================
153000   %==   SUBROUTINE TO EXECUTE XMSG MONITOR-CALL FOR TRANSFER ROUTINE    ===
153000   %=======================================================================
153000
153000   EGXMS: A=:9AREG;A:="SRSTR"=:"DRIVER";A:=XTBLK:=:L=:"MRETA":=9AREG
153007          *MON 2XMSG;JMP I (WT11
153011          IF T<0 THEN
153013              IF T+23=0 THEN T:=XMTRE=:CURMTY;T:=17=:CURFU;GO FAR LOGOF FI
153023              T-23;CALL XERR
153025          FI
153025          GO MRETA
153026
153026   %=======================================================================
153026   %======  SUBROUTINE TO EXECUTE XMSG MONITOR-CALL FOR SRQ ROUTINE  =======
153026   %=======================================================================
153026   SCALL: A=:9AREG;A:="SRSTR"=:"DRIVER";A:=SXTBL:=:L=:"SRETA";=9AREG
153035          *MON 2XMSG;JMP I (WT11
153037          IF T<0 THEN
153041               IF T+23=0 GO ENSIN
153044               T-23;CALL XERR
153046          FI
153046          GO SRETA
153047
153047   *)FILL
153066
153066   %=======================================================================
153066   %====            SUBROUTINE TO REMOVE USER FROM TABLES              ======
153066   %=======================================================================
153066    TRIPLE  GTREG                          % SAVELOCATION FOR TAD REG
153071    INTEGER GXREG                          % SAVELOCATION FOR X REG
153072
153072   TREMO: TAD=:GTREG;X=:GXREG                                              % SAVE REGISTERS
153074          X:=CGPUS;0=:UMESS(X)                                             % CLEAR MESSAGE TABLE ENTRY
153076          A:=X SHZ 1 =:X;A:=0=:D;AD=:UENV(X)                               % CLEAR USER/TERMINAL ENTRY
153104          A:=USCONT -1 =:USCONT                                            % UPDATE USER COUNT
153107          TAD:=GTREG;X:=GXREG;EXIT
153112
153112   %=======================================================================
153112   %====     SUBROUTINE TO WAIT FOR INERRUPT AND CHECK FOR TIMEOUT      ======
153112   %=======================================================================
153112
153112   GTSUB: T=:9TREG;T:=GPRUN; IF T =0 THEN T:=-1 ELSE T:=TTMR FI
153121          T=:TMR                                                           % START TIMER
153122          T:=L=:"GRETA";A=:9AREG;A:=D=:9DREG;X=:9XREG;CALL ID11            % SAVE REGISTERS AND GIVE UP LEVEL
153131          A:=TMR;IF A=0 THEN                                               % CHECK IF TIMEOUT
153133           0=:GPBFL;0=:ACTIV;0=:DINPT
153136           T:=GPRUN; IF T =0 THEN T:=-1 ELSE T:=TTMR FI
153144           T=:TMR;T:=3
153146   RSATO:  CALL TRNSF;CALL FATER;GO WAIT3;GO TORET                          % CLEAR CONTROLLER
153152   WAIT3:  T=:9TREG;CALL ID11
153154          A:=TMR;IF A=0 THEN 0=:GPBFL;T:=105;CALL FATER FI                 % IF TIMEOUT AFTER DEV. CLEAR ABORT G
153161          T:=9TREG;0=:TMR;GO RSATO
153164   TORET:  T:=-1=:9TREG
153166          FI
153166   GTRET: 0=:TMR;T:=9TREG;A:=9DREG=:D;A:=9AREG;X:=9XREG;GO GRETA
153175
153175   *)FILL
153177
153177   %=======================================================================
```

```
153177    %======= ROUTINE TO STOP GPIB SYSTEM ====================================
153177    %-----------------------------------------------------------------------
153177
153177    GPIBH: 0=:GPRUN;GO STCON
153201
153201    %-----------------------------------------------------------------------
153201    %===     ROUTINE FOR NONFATAL GPIB SYSTEM ERRORS     ====================
153201    %=======================================================================
153201
153201    NFERR: X:=L;A:=CURFU;CALL 9ERR(#93);L:=X;EXIT
153207
153207    %=======================================================================
153207    %===     ROUTINES FOR FATAL GPIB SYSTEM ERRORS       ====================
153207    %-----------------------------------------------------------------------
153207
153207    XRERR: T-1;A:=400;T+A;GO FATER                                    % XROUT ERROR
153213    XSERR: T:=A                                                       % XMSG SUBROUTINE ERROR
153214    XERR:  T-.;A:=300;T+A                                             % XMSG ERROR
153217    FATER: A:=L-1=:ERADD;A:=GPRUN                                     % GPIB ERROR
153223           IF A = 0 THEN T=:GPRUN;GO STCON FI
153226           A:=GPIBNO;AD SHZ -10;A -60 SHZ 3;A:=:D
153233           A SHZ -10; A -60;A+D;CALL 9ERR(#92)
153240           A:=CGPIM;IF A=0 GO STCON
153242           T=:CURFU;A:=32=:D;A:=B+"CURFU";X:=0;T:=XFWRI;CALL EGXMS   % STORE STATUS IN MESSAGE
153252           T:=XFSND BONE XFSEC;X:=CPORT;AD:=CURMN;CALL EGXMS         % RETURN MESSAGE
153257    STCON: -3=:TMR                                                   % START TIMER
153261           T:=XFDCT;A:=XTBLK=:L;*MON 2XMSG;JPL I (ID11               % DISCONNECT FROM XMSG SYSTEM
153266           T:=XFDCT;A:=SXTBL=:L
153271           IF A+1=0 THEN GO RSXTB FI;*MON 2XMSG                     % IF SRQ PORT OPENED, DISCONNECT FROM
153275    RSXTB: 0=:XTBLK=:SXTBL=:GPRUN;GO WT11
153301
153301
153301
153301    @MAC

)9SCLC
153301    %============================ GPIBD ======================================
153301    %
153301    %       DRIVER FOR GPIB BUS CONTROLLER
153301    %       CALLING SEQUENCE :
153301    %                           JPL I (DGPIB
153301    %                           JMP ERROR     % ERROR RETURN
153301    %                           JMP BUSY      % BUSY RETURN
153301    %                           JMP FINISH    % OPERATION FINISHED
153301    %
153301    %=======================================================================
153301    % PARAMETERS:
153301    %               T-REGISTER  : FUNCTION CODE
153301    %               B-REGISTER  : DATA FIELD
153301    %               X-REGISTER  : NO. OF BYTES ( FUNC 14)
153301    % RETURN :
153301    %               T-REGISTER  : STATUS
153301    %=======================================================================
153301    % FUNCTION CODES:
153301    %                       00 : DMA INPUT
153301    %                       01 : DMA OUTPUT
153301    %                       02 : READ STATUS
153301    %                       03 : CLEAR DEVICE
153301    %                       04 : READ PIO
153301    %                       05 : WRITE PIO
```

```
153301  %                       06 : ENABLE SRQ INTERRUPT
153301  %                       07 : DISABLE ALL INTERRUPTS
153301  %                       10 : SERIAL POLL
153301  %                       11 : PARALLELL POLL
153301  %                       12 : READ SYSTEM DEVICE LIST
153301  %                       13 : READ USER DEVICE LIST
153301  %                       14 : SEND CONTROL STRING WITHOUT DEVICE LIST
153301  %                       15 : SEND COMMAND BYTE WITH DEVICE LIST
153301  %                       16 : RUN MICROPROGRAM CONTROLLER TEST
153301  %=============================================================================
153301
153301  % REGISTER DEFINITION:
153301
153301  RDMEM=0
153301  LDMEM=1
153301  RDAT= 2
153301  LDAT= 3
153301  RSTAT=4
153301  LCONT=5
153301  LWORC=7
153301
153301  %=============================================================================
153301
153301  % GPIB CONTROLLER FUNCTION CODES
153301
153301  DMAUT=142
153301  EXSP= 102
153301  PIOUT=144
153301  PIOIN=145
153301  LSYSD=106
153301  LUSDV=104
153301  MTCOD=167
153301
153301  %=============================================================================
153301
153301  GPIBD, STF    I (TSAV
153302         STX    I (XSAV
153303         COPY   SL DX
153304         STX    I (LSAV
153305         COPY   ST DA
153306         AAA    -3
153307         JAZ    CLRDV            % CHECK IF FUNCTION=CLEAR DEVICE
153310         LDT    HDEV,B
153311         AAT    RSTAT
153312         IOXT
153313         STA    I (SSTAT
153314         LDA    I (TSAV
153315         AAA    -2               % CHECK IF FUNCION = READ STATUS
153316         JAZ    FINEX
153317         LDA    I (SSTAT
153320         BSKP   ZRO 40 DA
153321         JMP    ERREX
153322  %      BSKP   ONE 20 DA
153322         JMP    I (IFRDY
153323
153323         % BUSY EXIT
153323
153323  BUSEX, SAA    1
153324         STA    GPBFL,B
153325         LDA    I (LSAV
```

```
153326          COPY   SA DL
153327          LDF    I (TSAV
153330          LDX    I (XSAV
153331          EXIT   AD1
153332
153332          % CLEAR DEVICE WITHOUT TEST
153332
153332  CLRDV,  LDA    GPBFL,B
153333          JAF    RRES
153334          LDT    HDEV,B
153335          AAT    LCONT
153336          SAA    20
153337          IOXT
153340          SAA    3
153341          IOXT
153342          JMP    BUSEX
153343  RRES,   LDT    HDEV,B
153344          AAT    RDAT
153345          IOXT
153346          COPY   SA DT
153347          SUB    I (STMAS
153350          JAF    CEREX
153351
153351          % OPERATION FINISHED
153351
153351  FINEX,  LDA    I (LSAV
153352          COPY   SA DL
153353          LDD    I (ASAV
153354          LDT    I (SSTAT
153355          STZ    GPBFL,B
153356          STZ    ACTIV,B
153357          COPY   SL DL AD1
153360          EXIT   AD1
153361
153361          % ILLEGAL FUNCTION CODE OR
153361          % CORE ADDRESS REGISTER ERROR
153361
153361  ILFUC,  SAT    100
153362          JMP    *+2
153363  CAERR,  SAT    101
153364          LDA    I (LSAV
153365          COPY   SA DL
153366          STZ    GPBFL,B
153367          STZ    ACTIV,B
153370          EXIT
153371
153371  )FILL
153410
153410          % ERROR EXIT
153410
153410  ERREX,  LDT    HDEV,B
153411          AAT    RDAT
153412          IOXT
153413          AND    I (STMAS         % REMOVE USER NO.
153414          COPY   SA DT
153415  CEREX,  LDA    I (LSAV
153416          COPY   SA DL
153417          LDD    I (ASAV
153420          LDX    I (XSAV
153421          STZ    GPBFL,B
```

```
153422          STZ    ACTIV,B
153423          STZ    DINPT,B
153424          EXIT
153425
153425  )FILL
153431
153431          % INTERFACE READY
153431
153431  IFRDY,  LDA    I (TSAV
153432          COPY   SA DT
153433          AAA    -17
153434          JAP    ILFUC
153435          RADD   ST DP
153436          JMP    I (DMA           % OPERATION IS DMA INPUT
153437          JMP    I (DMA           % OPERATION IS DMA OUTPUT
153440          JMP    I (FINEX         % OPERATION IS READ STATUS
153441          JMP    I (CLRDV         % OPERATION IS CLEAR DEVICE (THIS BRANCH SHOULD NEVER BE USED)
153442          JMP    I (RPIO          % OPERATION IS READ PIO
153443          JMP    I (WPIO          % OPERATION IS WRITE PIO
153444          JMP    I (SRQEN         % OPERATION IS ENABLE SRQ INTERRUPT
153445          JMP    I (DISIN         % OPERATION IS DISABLE ALL INTERRUPTS
153446          JMP    I (POLL          % OPERATION IS SERIAL POLL
153447          JMP    I (POLL          % OPERATION IS PARALLEL POLL
153450          JMP    I (RDEVL         % OPERATION IS READ SYSTEM DEVICE LIST
153451          JMP    I (RDEVL         % OPERATION IS READ USER DEVICE LIST
153452          JMP    I (WCOMD         % OPERATION IS SEND CONTROL STRING (WITHOUT DEVICE LIST)
153453          JMP    I (WCOTD         % OPERATION IS SEND CONTROL STRING (WITH DEVICE LIST)
153454          JMP    I (MTST          % OPERATION IS RUN MICROPROGRAN TEST
153455
153455  )FILL
153472
153472          % DMA OPERATION
153472
153472  DMA,    LDA    GPBFL,B
153473          JAZ    INDMA
153474          LDA    ACTIV,B
153475          JAZ    DMACT
153476          LDT    HDEV,B
153477          IOXT
153500          LDT    I (SSTAT
153501          LDX    I (DSAV
153502          RSUB   SX DA
153503          SHA    ZIN 1
153504          BSKP   ZRO 130 DT
153505          RDCR   DA
153506          STA    CURBC,B
153507          LDA    I (TSAV
153510          SKP    IF DA UEQ 0
153511          TRR    CCLR          % CLEAR CACHE
153512          JMP    I (FINEX
153513
153513          % ENABLE SRQ INTERRUPT
153513
153513  SRQEN,  SAA    0
153514          BSET   ONE 60 DA
153515          LDT    HDEV,B
153516          AAT    LCONT
153517          IOXT
153520          JMP    I (FINEX
153521
```

```
153521          % DISABLE INTERRUPTS
153521
153521  DISIN,  SAA     0
153522          LDT     HDEV,B
153523          AAT     LCONT
153524          IOXT
153525          JMP     I (FINEX
153526
153526          % ACTIVATE DMA TRANSFER
153526
153526  DMACT,  LDD     CURAD,B
153527          LDT     HDEV,B
153530          AAT     LOMEM
153531          IOXT
153532          COPY    SD DA
153533          IOXT
153534          LDA     CURBC,B
153535          SHA     ZIN SHR 1
153536          BSKP    ZRO SSM
153537          AAA     1
153540          LDT     HDEV,B
153541          AAT     LWORC
153542          IOXT
153543          SAA     7
153544          LDT     I (TSAV
153545          BSKP    ONE 00 DT
153546          BSET    ONE 130 DA
153547          LDT     HDEV,B
153550          AAT     LCONT
153551          IOXT
153552  ACTEX,  SAA     1
153553          STA     ACTIV,B
153554          JMP     I (BUSEX
153555
153555          % INIT DMA TRANSFER
153555
153555  INDMA,  JPL     I (SUSER
153556          SAA     DMAUT
153557          LDX     I (TSAV
153560          BSKP    ONE 00 DX
153561          AAA     -1
153562          IOXT
153563          LDA     DINPT,B
153564          BSET    ONE 00 DA
153565          BSKP    ONE 00 DX
153566          STA     DINPT,B
153567          JPL     I (SNOBY
153570          JPL     I (SINST
153571          JMP     I (ENINT
153572
153572          % PROGRAMMED I/O WRITE
153572
153572  WPIO,   LDA     GPBFL,B
153573          SKP     IF DA EQL 0
153574          JMP     I (FINEX
153575          JPL     I (SUSER
153576          SAA     PIOUT
153577          IOXT
153600          JPL     I (SNOBY
153601          JPL     I (SINST
```

```
153602          JPL    WGPID
153603          JMP    I (ENINT
153604
153604          % WRITE DATA
153604
153604  WGPID,  LDD    CURAD,B
153605          STD    I (PADDR
153606          LDX    CURBC,B
153607  WNXTW,  LDD    I (PADDR
153610          EXAM
153611          RINC   DD
153612          COPY   SA DA ADC
153613          STD    I (PADDR
153614          RCLR   DA
153615          COPY   ST DD
153616          SAD    ZIN 10
153617          LDT    HDEV,B
153620          AAT    LDAT
153621          IOXT
153622          RDCR   DX
153623          JXZ    NODAT
153624          RCLR   DA
153625          SAD    ZIN 10
153626          IOXT
153627          RDCR   DX
153630          SKP    IF DX EQL 0
153631          JMP    WNXTW
153632  NODAT,  EXIT
153633
153633  )FILL
153645
153645          % PROGRAMMED I/O READ
153645
153645  RPIO,   LDA    GPBFL,B
153646          JAZ    RPINI
153647          LDA    FIRST,B
153650          JAZ    RGPDA
153651
153651          % READ BYTECOUNT FROM INTERFACE AND INIT WORCOUNT AND DATAPOINTER
153651
153651  RBCON,  LDT    HDEV,B
153652          AAT    RDAT
153653          IOXT
153654          STA    CURBC,B
153655          LDT    GPUZI,B
153656          SKP    IF DT MLST SA
153657          JMP    BCOK
153660          SAT    106
153661          JMP    I (CEREX
153662  BCOK,   SHA    ZIN SHR 1
153663          BSKP   ZRO SSM
153664          AAA    1
153665          STA    PIOWC,B
153666          SKP    IF DA UEQ 0
153667          JMP    I (FINEX
153670          STZ    FIRST,B
153671          STZ    PIODP,B
153672          SAA    1
153673          STA    DINPT,B
153674          SAA    3
```

```
153675          LDT   HDEV,B
153676          AAT   LCONT
153677          IOXT
153700          JMP   I (BUSEX
153701
153701          % INIT PIO TRANSFER
153701
153701  RPINI,  JPL   SUSER
153702          SAA   PIOIN
153703          STA   FIRST,B
153704          IOXT
153705          JPL   SNOBY
153706          JPL   SINST
153707          JMP   ENINT
153710
153710
153710  SUSER,  LDA   CGPUS,B
153711          LDT   HDEV,B
153712          AAT   LDAT
153713          IOXT
153714          EXIT
153715
153715
153715  SNOBY,  LDA   CURBC,B
153716          IOXT
153717          SHA   ZIN SHR 10
153720          IOXT
153721          EXIT
153722
153722
153722  SINST,  SAX   0
153723          SAA   -10
153724          STA   I (ICONT
153725  NXINS,  LDA   INSTA,B,X
153726          SAD   ZIN SHR 10
153727          IOXT
153730          BSKP  ZRO 70 DA
153731          JMP   LINS
153732          SAD   ZIN 10
153733          IOXT
153734          BSKP  ZRO 70 DA
153735          JMP   LINS2
153736          AAX   1
153737          MIN   I (ICONT
153740          JMP   NXINS
153741          JMP   IERR
153742  LINS,   BSKP  ONE 60 DA
153743          JMP   IFI
153744          SAD   ZIN 10
153745          IOXT
153746          JMP   IFI
153747  LINS2,  BSKP  ONE 60 DA
153750          JMP   IFI
153751          AAX   1
153752          LDA   INSTA,B,X
153753          SAD   ZIN SHR 10
153754          IOXT
153755  IFI,    EXIT
153756  IERR,   SAA   70
153757          BSET  ONE 70 DA
```

```
153760          IOXT
153761          EXIT
153762
153762
153762  ENINT,  SAA    3
153763          BSET   ONE 100 DA
153764          LDT    HDEV,B
153765          AAT    LCONT
153766          IOXT
153767          JMP    I (BUSEX
153770
153770  )FILL
153774
153774          % READ DATA
153774
153774  RGPDA,   LDT    HDEV,B
153775          AAT    RDAT
153776          IOXT
153777          COPY   SA DT
154000          LDD    CURAD,B
154001          LDX    PIODP,B
154002          RADD   SX DD
154003          COPY   SA DA ADC
154004          DEPO
154005          AAX    1
154006          STX    PIODP,B
154007          LDA    PIOWC,B
154010          AAA    -1
154011          STA    PIOWC,B
154012          JAZ    FINI
154013          SAA    3
154014          LDT    HDEV,B
154015          AAT    LCONT
154016          IOXT
154017          JMP    I (BUSEX
154020  FINI,   JMP    I (FINEX
154021
154021  )FILL
154023
154023          % SEND COMMAND BYTES WITHOUT DEVICELIST
154023
154023  WCOMD,  LDA    GPBFL,B
154024          SKP    IF DA EQL 0
154025          JMP    I (FINEX
154026          JPL    SUSER
154027          JPL    I (WGPID
154030          JMP    WINT
154031
154031          % SEND COMMAND WITH DEVICELIST
154031
154031  WCOTD,  LDA    GPBFL,B
154032          SKP    IF DA EQL 0
154033          JMP    I (FINEX
154034          JPL    SUSER
154035          LDD    CURAD,B
154036          COPY   ST DX
154037          EXAM
154040          COPY   ST DA
154041          COPY   SX DT
154042          IOXT
```

```
154043          JPL    SINST
154044  WINT,   SAA    2
154045          BSET   ONE 100 DA
154046          LDT    HDEV,B
154047          AAT    LCONT
154050          IOXT
154051          JMP    I (BUSEX
154052
154052
154052
154052          % READ POLLSTATUS
154052
154052  POLL,   LDA    GPBFL,B
154053          JAZ    POINI
154054          LDT    HDEV,B
154055          AAT    RDAT
154056          IOXT
154057          STA    SSTAT
154060          JMP    I (FINEX
154061
154061          % INIT SERIAL OR PARALLEL POLL
154061
154061  POINI,  JPL    I (SUSER
154062          SAA    EXSP
154063          LDT    TSAV
154064          BSKP   ZRO 00 DT
154065          AAA    1
154066          LDT    HDEV,B
154067          AAT    LDAT
154070          IOXT
154071          SAA    3
154072          LDT    HDEV,B
154073          AAT    LCONT
154074          IOXT
154075          JMP    I (BUSEX
164076
154076          % READ SYS/USER DEVICELIST
154076
154076  RDEVL,  LDA    GPBFL,B
154077          JAZ    RDINI
154100          LDA    FIRST,B
154101          JAF    BCON
154102          JMP    I (RGPDA
154103  BCON,   JMP    I (RBCON
154104  RDINI,  LDA    CGPUS,B
154105          LDT    HDEV,B
154106          AAT    LDAT
154107          IOXT
154110          SAA    LSYSD
154111          LDX    TSAV
154112          BSKP   ZRO 00 DX
154113          SAA    LUSDV
154114          IOXT
154115          SAA    3
154116          STA    FIRST,B
154117          LDT    HDEV,B
154120          AAT    LCONT
154121          IOXT
154122          JMP    I (BUSEX
154123
```

```
154123  MTST,   LDA    GPBFL,B
154124          JAF    RDCDE
154125          LDT    HDEV,B
154126          AAT    LDAT
154127          SAA    MTCOD
154130          IOXT
154131          AAT    -LDAT+LCONT
154132          SAA    3
154133          IOXT
154134          JMP    I (BUSEX
154135  RDCDE,  LDT    HDEV,B
154136          AAT    RDAT
154137          IOXT
154140          STA    SSTAT
154141          SHA    ZIN 10
154142          SKP    IF DA EQL 0
154143          JMP    I (FINEX
154144          JMP    I (ERREX
154145  )FILL
154154
154154
154154  TSAV,   0
154155  ASAV,   0
154156  DSAV,   0
154157  BSAV,   0
154160  XSAV,   0
154161  LSAV,   0
154162  SSTAT,  0
154163  ICONT,  0
154164  STMAS,  377
154165  PADDR,  0;0
154167
154167  )KILL RDMEM LOMEM RDAT LDAT RSTAT LCONT LWORC DMAUT EXSP PIOUT
154167  )KILL PIOIN LSYSD LUSDV MTCOD BUSEX CLRDV RRES FINEX ILFUC CAERR
154167  )KILL ERREX CEREX IFRDY DMA SRQEN DISIN DMACT ACTEX INDMA WPIO
154167  )KILL WGPID WNXTW NODAT RPIO RBCON BCOK RPINI SUSER SNOBY SINST
154167  )KILL NXINS LINS LINS2 IFI IERR ENINT RGPDA FINI WCOMD WCOTD WINT
154167  )KILL POLL POINI RDEVL BCON RDINI MTST RDCDE TSAV ASAV DSAV BSAV
154167  )KILL XSAV LSAV SSTAT ICONT STMAS PADDR
154167  )9RCLC
)9SLPL154167
154167   %=============================================================================
154167   %===============  ROUTINE TO PERFORM TRANSFER TO GPIB DRIVER  ==============
154167   %===============  ACKTIVATED FROM ABSTRANS                    ==============
154167   % PARAMETER LIST:   PAR, IFUNC          % DUMMY
154167   %                    DMEM            % PHYS. MEMORY ADDRESS  FOR DMA
154167   %                    DINSI           % PHYS. MEMORY ADDRESS TO DATA ARRAY
154167   %                    DINSO           % PHYS. MEMORY ADDRESS FOR RETURN DATA ARRAY
154167   %=============================================================================
154167   INTEGER LOOP,SAVX
154171   GPITR: IF GPRUN = -1 GO GPAE1                          % XMSG DRIVER RUNNING
154175          1=:GABSF                                        % SET ABSTR FLAG
154177          X:=:B
154200          CALL PICKYLPAR                                  % GET DINSI IN AD
154201          X:=:B; X=:SAVX
154203          A=:T;D=:X; "CURFU"=:D
154207          -15=:LOOP; FOR LOOP DO
154211            *LDATX; SWAP SD DX; STA ,B,X; SWAP SD DX    % MOVE DATA ARRY TO DATAFIELD
154215            X+1;D+1
154217          OD
```

```
154221          X:=SAVX; X:=:B
154223          CALL PICKFPAR;  AD=:X.CURAD                    % GET DMEM
154225          CALL PICKXLPAR; AD=:X.DINSO                    % GET DINSO
154227          B:=X
154230          IF CURFU = 6 OR = 7 OR = 10 OR > 22 GO GPAE1
154245   GPATR: T:=CURFU; X:=CURBC;
154247          CALL TRANSF; GO GPABY; GO GPAFI; GO GPAFI;

% **** ERROR AT: 154247 **** RANGE EXCEEDED
154253
154253   GPABY: CALL ID11; GO GPATR
154255   GPAFI: IF RTRES >< 0 THEN
154257            AD:=DINSO; A=:T;D=:X; "CURFU"=:D             % GET ADDRESS TO DATA ARRAY
154264            -15=:LOOP; FOR LOOP DO                       % RETURN DATA ARRAY
154266              *SWAP SD DX; LDA ,B,X; SWAP SD DX; STATX;
154272              X+1; D+1;
154274            OD
154276            GO L1
154277          FI
154277   GPAE1: -1=:HSTAT
154301   L1:    0=:GABSF
154302          CALL RTACT
154303          GO GPABY
154304   RBUS
154312   *"
"154312    *"BOCTO
"154312    %=================================================================%
154312   %       O C T O B U S _ D R I V E R                                 %
154312   %=================================================================%
154312   SUBR OCTOBUS, STARC, RKTASK
154312   % --- SYSTEM CONSTANTS
154312
154312   SYMBOL MAXTASK = 40, OBOXSI = 14
154312   SYMBOL BSETO = 174205                % BSET ONE DA
154312   SYMBOL INTRR = 153600                % IRR
154312   SYMBOL INTRW = 153400                % IRW
154312
154312   SYMBOL FALS = 0, TRUE = 1, NIL = 0
154312   % TYPE CB=ENUMERATION(DATAM,CONTROL)
154312   SYMBOL CONTROL=17
154312   % TYPE BB=ENUMERATION(DIRMINI,BROADCAST)
154312   SYMBOL DIRMINI=0,BROADCAST
154312   % TYPE FNC=ENUMERATION(MINIPCK,HWSTTS,MULPCK,HWCNT)
154312   SYMBOL MINIPCK=0,HWSTTS,MULPCK,HWCNT
154312   % HARDWARE CONTROL MESSAGE
154312   @ICR;
154312   SYMBOL STPR=0,REPR,RUNPR,STOPPR,SINGSTEP,
154312          RESSISTEP,SETDIAG,RDIAG,
154312          H10,H11,H12,H13,H14,H15,H16,H17,
154312          POWER,PWERISUP,WHOCLOCK,CLOMA,H24,H25,H26,H27,
154312          H30,H31,H32,H33,H34,H35,PWERUP,PWERDOWN;
154312   @CR;
154312   % --- OCTOBUS MESSAGE
154312   % TYPE OCTO = RECORD PACK
154312   %         CB : C                              % CONTROL OR DATA
154312   %         BB : B                              % BROADCAST OR DIRECT MINIPACKET
154312   %         INTEGER RANGE (0:77B) : SOURCE  % CPU NUMBER FOR MINIPACKEC SOURCE
154312   %         INTEGER RANGE (0:77B) : DESTINI=SOURCE
154312   %         INTEGER RANGE (0:77B) : NOD_TYP=SOURCE % FOR BROADCAST
154312   %         BOOLEAN : E                         % EMERGENCY
```

```
154312   %         FNC : F                          % FUNCTION
154312   %         INTEGER RANGE (0:37B) : OPCODE  % IF F = MINIPCK & C = CONTROL
154312   %         INTEGER RANGE (0:37B) : N = OPCODE % IF F = MULPCK & C = CONTROL
154312   %         HWC : BASCNT = OPCODE           % BASIC CONTROL FUNCTIONC
154312   %         INTEGER RANGE (0:377B) : INFORMATION = E % IF F = MINIPCK & C = DATA
154312   %         INTEGER2 : MSK = OPCODE
154312   % ENDRECORD
154312   % --- OCTOBUS FRAME
154312   % TYPE FRAME = RECORD PACK
154312   %         INTEGER RANGE (0:17B) : PRIORITY % PRIORITY
154312   %         INTEGER RANGE (0:77B) : DESTINI % DESTINATION
154312   %         OCTO : BODY
154312   %         INTEGER RANGE (0:3) : PARITY     % PARITY (NO. OF "1"S IS COUNTED, 2 LEAST SIGN. BITS!)
154312   %         INTEGER RANGE (0:3) : ACKNOW     % ACKNOWLEDGE FROM DEST. NODE
154312   % ENDRECORD
154312   % --- RECEIVE STATUS WORD
154312   % TYPE RECEIVSTA = RECORD PACK
154312   %         INTEGER RANGE (0:3)    : RES1     % NOT USED
154312   %         INTEGER RANGE (0:77B) : STATNO   % OWN STATION NUMBER
154312   %         INTEGER RANGE (0:17B) : RSW2     % NOT USED
154312   %         BOOLEAN : FIFO_RFTR              % RECEIVE FIFO READY FOR TRANSMIT
154312   %                                          % (NOT EMPTY). RECEIVED DATA IN FIFO
154312   %         BOOLEAN : FIFO_FULL              % RECEIVE FIFO FULL (MUST BE EMPTIED)
154312   %         BOOLEAN : RS1                    % NOT USED
154312   %         BOOLEAN : RINTEN                 % RECEIVE INTERUPT ENABLED
154312   %         INTEGER2 : MSK = RINTEN
154312   % ENDRECORD
154312   SYMBOL    FIFORFTR=3                  % RECEIVE FIFO READY FOR TRANSMIT
154312                                         % (NOT EMPTY). RECEIVED DATA IN FIFO
154312   SYMBOL    FIFOFULL=2                  % RECEIVE FIFO FULL (MUST BE EMPTIED)
154312   SYMBOL    RS1=1                       % NOT USED
154312   % SYMBOL    RINTEN=0                  % RECEIVE INTERUPT ENABLED
154312
154312   % --- RECEIVE CONTROL WORD
154312   % TYPE RECEIVCNT = RECORD PACK
154312   %         INTEGER RANGE (0:17B) : RCW      % NOT USED
154312   %         INTEGER RANGE (0:77B) : STATNO  % OWN STATION NUMBER
154312   %         BOOLEAN : READALL
154312   %         BOOLEAN : CLEARDEV               % CLEAR DEVICE
154312   %         BOOLEAN : RC3                    % LOCAL LOOP (NOT USED)
154312   %         BOOLEAN : RC1,RC2                % NOT USED
154312   %         BOOLEAN : RINTEN                 % RECEIVE INTERUPT ENABLE
154312   %         INTEGER2 : MSK = RINTEN
154312   % ENDRECORD
154312
154312   SYMBOL    READALL=5
154312   SYMBOL    CLEARDEV=4                  % CLEAR DEVICE
154312   SYMBOL    RC3=3                       % LOCAL LOOP (NOT USED)
154312   SYMBOL    RC1=1,RC2                   % NOT USED
154312   SYMBOL    RINTEN=0                    % RECEIVE INTERUPT ENABLE
154312
154312   % --- TRANSMIT STATUS WORD
154312   % TYPE TRNSSTA = RECORD PACK
154312   %         BOOLEAN : MASTER
154312   %         INTEGER RANGE (0:77B) : TSW      % NOT USED
154312   %         BOOLEAN : PERROR                 % TRANSMITTED WITH PARITY ERROR ON SINGLE RECEIVER
154312   %         BOOLEAN : BUSYDEST               % TRANSMITTED WITH BUSY DESTINATION (FIFO FULL)
154312   %         BOOLEAN : NO_PRESENT             % TRANSMITTED WITH NO RETRIES
154312   %         BOOLEAN : RETRY_TIMOUT           % TRANSMITTED RETRY TIMEOUT
154312   %         BOOLEAN : TSERROR                % TRANSMIT ERROR
```

```
154312   %         BOOLEAN : READY                  % TRANSMIT FIFO EMPTY
154312   %         BOOLEAN : ACTIV
154312   %         BOOLEAN : TS1                    % NOT USED
154312   %         BOOLEAN : TINTEN                 % TRANSMIT INTERUPT ENABLED
154312   %         INTEGER2 : MSK = TINTEN
154312   % ENDRECORD
154312
154312   SYMBOL    PERROR=10                  % TRANSMITTED WITH PARITY ERROR ON SINGLE RECEIVER
154312   SYMBOL    BUSYDEST=7                 % TRANSMITTED WITH BUSY DESTINATION (FIFO FULL)
154312   SYMBOL    NOPRESENT=6                % TRANSMITTED WITH NO RETRIES
154312   SYMBOL    RETRYTIMOUT=5              % TRANSMITTED RETRY TIMEOUT
154312   SYMBOL    TSERROR=4                  % TRANSMIT ERROR
154312   SYMBOL    TFIEM=3                    % TRANSMIT FIFO EMPTY
154312   SYMBOL    ACTIV=2
154312   SYMBOL    TS1=1                      % NOT USED
154312   SYMBOL    TINTEN=0                   % TRANSMIT INTERUPT ENABLED
154312
154312   % --- TRANSMIT CONTROL WORD
154312   % TYPE TRNSCNT = RECORD PACK
154312   %         INTEGER RANGE (0:177B) : RETRY  % RETRY COUNTER, -1=0 RETRIES, 0=256
154312   %         BOOLEAN : TC7,TC8               % NOT USED
154312   %         BOOLEAN : FIFO_CL               % CLEAR FIFO
154312   %         BOOLEAN : MASTER_RSET           % RESET MASTER
154312   %         BOOLEAN : CLEARDEV              % CLEAR DEVICE
154312   %         BOOLEAN : TC3                   % NOT USED
154312   %         BOOLEAN : ACTIVATE              % RESET TRANSMIT RFT
154312   %         BOOLEAN : TC1                   % NOT USED
154312   %         BOOLEAN : TINTEN                % TRANSMIT INTERUPT ENABLE
154312   %         INTEGER2 : MSK = TINTEN
154312   % ENDRECORD
154312
154312   SYMBOL    TC7=7,TC8                  % NOT USED
154312   SYMBOL    FIFOCL=6                   % CLEAR FIFO
154312   SYMBOL    MASTERRSET=5               % RESET MASTER
154312   % SYMBOL   CLEARDEV=4                % CLEAR DEVICE
154312   SYMBOL    TC3=3                      % NOT USED
154312   % SYMBOL   ACTIVATE=2                % RESET TRANSMIT RFT
154312   SYMBOL    TC1=1                      % NOT USED
154312   % SYMBOL   TINTEN=0                  % TRANSMIT INTERUPT ENABLE
154312
154312   % --- STANDARD SINTRAN - III - DATAFIELD PART - ONE FOR EACH RT PROGRAM
154312   % --- DEVICE DEPENDANT DATAFIELD - ONE FOR EACH OCTOBUS
154312   %     OVERLAPPED WITH THE LOCAL-STACK AREA OF THE ROUTINE "OCTOBUS"!
154312   DISP -75
154312   INTEGER   RECE                     % INDICATES OCTO-DRIVER CALLED OR BY INTERUPT
154312   INTEGER   DLEVL                      % RETURN LEVEL FOR DIRECT TASK STATUS - A/D REG
154312   INTEGER   PB                         % ACTUAL TASKS POST-BOX ADDRESS
154312   INTEGER   RETUR                      % RETURN ADDRESS FROM SUBROUTINE RKTASK
154312   INTEGER   UTILL
154312   % BYTES ARRAY : MULTIBYTES(0:63,0:4)    % MULTI BYTES SAVE ARRAY
154312   % BOOLEAN ARRAY : MULBY(0:63)           % TRUE IF MULTI-BYTE MESSAGE RECEIVED
154312   % INTEGER2 ARRAY : NOOFBYTES(0:63)      % NUMBER OF BYTES IN MULTI-BYTE MESSAGE
154312   INTEGER ARRAY TDATA(MAXTASK)
154312   INTEGER   SLOTNO                   % T-REG - TASK NUMBER
154312   INTEGER   CNTROLINFO=SLOTNO        % CONTROL INFORMATION
154312   INTEGER   OFNC                     % A-REG - FUNCTION
154312   INTEGER   CPU                      % D-REG - DESTINATION CPU
154312   INTEGER   PPREG=CPU                % P-REG - WHERE DIRECT TASK IS TO BE STARTED
154312   INTEGER   TRENA=CPU                % DISABLE = 0 / ENABLE = 1
154312   INTEGER   ACBOX                    % ACTUAL TASK (POST-BOX) NUMBER
```

```
154312    INTEGER   TD                    % CURRENT TRANSMIT DATA (DATA TO SEND)
154312    INTEGER   TS                    % TRANSMIT STATUS WORD
154312    INTEGER   TC                    % TRANSMIT CONTROL WORD
154312    INTEGER   RD                    % RECEIVED DATA WORD
154312    INTEGER   RS                    % RECEIVED STATUS WORD
154312    INTEGER   RC                    % RECEIVED CONTROL WORD
154312    INTEGER   WQUEUE                % HEAD OF TRANSMIT QUEUE- INIT = NIL
154312    INTEGER   TOTTRANS              % TOTAL NUMBER OF TRANSMITTED MESSAGES
154312    INTEGER   TOTRECEIV             % TOTAL NUMBER OF RECEIVED MESSAGES
154312    INTEGER   LASTSLOT              % PREVIOUS SENT MESSAGE - INIT = -1
154312    INTEGER   POSTBOX               % ONE POST-BOX FOR EACH TASK (RT OR DIRECT)
154312    INTEGER   OWNCPU                % CPU NUMBER OF "THIS" MACHINE
154312    INTEGER   BSTDA
154312    PSID
154312
154312    %  SINTRAN III STANDARD PART
154312
154312    % TYPE OCTO_MODE = RECORD PACK
154312    %         BOOLEAN : WTRANS              % WAIT FOR TRANSMIT MESSAGE (ERROR INTERUPT)
154312    %         BOOLEAN : WKICK               % START ON "KICK"?
154312    %         BOOLEAN : REP                 % KICKED BEFORE WAIT-FOR-KICK IS DONE
154312    % ENDRECORD
154312    SYMBOL  WTMEI=17                % WAIT FOR TRANSMIT MESSAGE (ERROR INTERUPT)
154312    SYMBOL  WKICK=16                % START ON "KICK"?
154312    SYMBOL  REP=15                  % KICKED BEFORE WAIT-FOR-KICK IS DONE
154312    % --- POST BOX DESCRIPTION - ONE FOR EACH TASK (INCLUDED RT PROGRAM)
154312    DISP 0
154312    INTEGER  POCTO                  % POINTER TO RT PROGRAMS DATAFIELD
154312    INTEGER  LINK                   % WQUEUE LINK
154312    INTEGER  OMODE                  % OSTOBUS MODE
154312    INTEGER  PREG                   % START ADDR FOR DIRTASK ON DRIVER LEVEL
154312    INTEGER  BOXN                   % POST-BOX NUMBER
154312    INTEGER  KICKCNT                % "KICK" COUNTER
154312    INTEGER  TERRCNT                % TRANSMIT-ERROR COUNTER
154312    INTEGER  RETFUNC                % RETURN VALUE
154312    INTEGER  TRNSTAT                % TRANSMIT STATUS ON RETURN
154312    INTEGER  RECSTAT                % RECEIVE STATUS ON RETURN
154312    INTEGER  LEVEL                  % DIRECT TASK LEVEL
154312    INTEGER  RESERVELOC

% **** ERROR AT: 154312 **** ALREADY DEFINED RESER
154312    PSID
154312    % --- "IOX" NUMBERS
154312    SYMBOL RCIVD              = 0
154312    SYMBOL RCIVST             = 2
154312    SYMBOL RCIVCW             = 3
154312    SYMBOL TRNSD              = 5
154312    SYMBOL TRNSST             = 6
154312    SYMBOL TRNSCW             = 7
154312    % --- ERROR NUMBERS
154312    SYMBOL OKRET               = 0
154312    SYMBOL ILFUNC              = 53     % FUNCTION IN RECIVE DATA IS WRONG
154312    SYMBOL NORES               = 54     % NO RT-PROGRAM HAS RESERVED THE POST-BOX
154312    SYMBOL ILHFUNC             = 55     % ILLEGAL BASIC CONTROL FUNCTION
154312    SYMBOL MULERR              = 56     % MULTIBYTE MESSAGE WITH NO HEADER
154312    SYMBOL ILSLOT              = 57     % WRONG SLOT (TASK) NUMBER
154312    SYMBOL ILTSS               = 60     % WRONG TRANSMIT STATUS
154312    SYMBOL TRERR               = 61     % TRANSMIT ERROR
154312    SYMBOL PTRERR              = 62     % ERROR IN PREVIOUS TRANSMIT
154312    SYMBOL NOTTR               = 63     % CURRENT TRANSMIT NOT FINISHED
```

```
154312    SYMBOL PRTRANS              = 64      % PREVIOUS TRANSMIT NOT FINISHED
154312    SYMBOL NOKICK               = 65      % TASK IS NOT PREPARED FOR "KICK"
154312    SYMBOL SCPC                 = 66      % SEND CONTROL PACKAGE NOT IMPLEMENTED
154312    SYMBOL WFU                  = 67      % WRONG OCTOBUS FUNCTION IN TRANSMIT
154312    SYMBOL NOINIT               = 70      % NO "P-REG" IS INITIATED FOR DIRECT TASK
154312    SYMBOL HWST                 = 71      % HARDWARE STATUS
154312    SYMBOL MULP                 = 72      % MULTIBYTE PACKAGE
154312    SYMBOL MULPD                = 73      % MULTIBYTE PACKAGE DATA BYTE
154312    SYMBOL NORT                 = 74      % NOT LEGAL FOR RT-PROGRAMS
154312    SYMBOL NOTAS                = 75      % NOT LEGAL FOR TASKS
154312
154312    % --- "USER" FUNCTIONS
154312    SYMBOL          KICK=0                % "KICK" A TASK
154312    SYMBOL          IOW=1                 % HANG IN "IO-WAIT" FOR A "KICK"
154312    SYMBOL          RTA=2                 % ACTIVATED ONCE A "KICK" COMES
154312    SYMBOL          PKIC=3                % PREPARE A "P-REG" FOR THE "KICK"
154312    SYMBOL          SCPCK=4               % SEND CONTROL PACKET
154312    SYMBOL          RLSTA=5               % READ LAST STATUS
154312    SYMBOL          WHOAMI=6              % WHO AM I
154312
154312
154312    % *** OCTOBUS DRIVER ***
154312
154312    %=====================================================================%
154312    %=!     O C T O B U S                                                 %
154312    %=====================================================================%
154312
154312    %       PREPARE, TRANSMIT AND RECEIVE PART
154312    %       OVERLAPS SIN III OCTOBUS DATAFIELD!
154312    %
154312    %       T-REG - TASK NUMBER
154312    %             - CONTROL INFORMATION
154312    %       A-REG - FUNCTION
154312    %       D-REG - DESTINATION CPU
154312    %             -  P-REG DISABLE = 0 / ENABLE = 1
154312    %       P-REG - WHERE DIRECT TASK IS TO BE STARTED
154312
154312
154312    % ***  RECEIVE AND EXECUTE MESSAGES - "INPUT" PART OF THE OCTO-BUS ***
154312    OCTOBUS:
154312    * IOF; COPY SA DX; TRA PVL; ION
154316    A=:DLEVL; X=:A
154320    * STF SLOTN ,B                        % SAVE TAD-REG
154321    DLEVL /\ 170=:DLEVL
154324    BEGDO;
154324    T:=HDEV+RCIVST;* IOXT; STA RS,B       % READ RECEIVE STATUS
154330    IF RS NBIT FIFORFTR GO FAR WHIL       % WHILE RS.FIFO_RFTR
154333    TOTRECEIV + 1 =:TOTRECEIV
154336    I:=HDEV+RCIVD;* IOXT; STA RD,B        % READ RECEIVED DATA
154342    IF RD NBIT CONTROL GO FAR DATAB       % CONTROL MESSAGE
154345    IF RD /\ 140 SH -5 = MINIPCK THEN     % CHECK FUNCTION
154353       RD /\ 37 =:ACBOX
154356          A * OBOXSI + POSTBOX=:PB; OKRET=:PB.RETFUNC
154364          IF ACBOX > 17 THEN
154370             IF X.OMODE BIT WKICK THEN
154373                TRUE; CALL RKTASK;CALL 5ERET
154376                A:=PB.OMODE; A BZERO WKICK=:X.OMODE
154402             ELSE A:=PB.OMODE; A BONE REP=: X.OMODE % $* LDA NOKICK; CALL 9ERR(#99); 32460
154407             FI
154407          ELSE
```

```
154410            IF X.PREG >< 0 THEN A=:L
154413               RD /\ 37400 SH -10; L:=:P
154417            ELSE A:=NOINIT; CALL 9ERR(#99)
154423            FI
154423         FI
154423   ELSE IF A = HWSTTS THEN
154427         A:= HWST; CALL 9ERR(#99)
154432   ELSE IF A = MULPCK THEN
154436         A:= MULP; CALL 9ERR(#99)
154441         * JMP FILL1; )FILL; FILL1=*; )KILL FILL1
154454   %     TRUE=:MULBY(RD.SOURCE); RD.OPCODE=:NOOFBYTES(RD.SOURCE)
154454   ELSE IF A = HWCNT THEN
154460         IF RD /\ 37 = STPR THEN                          % CHECK FUNCTION
154465         ELSE IF A = REPR      THEN
154471         ELSE IF A = RUNPR        THEN
154475         ELSE IF A = STOPPR       THEN
154501         ELSE IF A = SINGSTEP     THEN
154505         ELSE IF A = RESSISTEP    THEN
154511         ELSE IF A = SETDIAG      THEN
154515         ELSE IF A = RDIAG        THEN
154521         ELSE IF A = POWER        THEN
154525         ELSE IF A = PWERISUP     THEN
154531         ELSE IF A = WHOCLOCK     THEN
154535         ELSE IF A = CLOMA        THEN
154541         ELSE IF A = PWERUP       THEN
154545         ELSE IF A = PWERDOWN     THEN
154551         ELSE A:= ILHFUNC
154553      FI;FI;FI;FI;FI;FI;FI;FI;FI;FI;FI;FI;FI;FI
154553      CALL 9ERR(#99)
154555   ELSE A:= ILFUNC; CALL 9ERR(#99)
154561   FI;FI;FI;FI
154561   GO ENDDO
154562   DATAB:                                                % MULTI BYTE DATA MESSAGE
154562   A:= MULPD; CALL 9ERR(#99)
154565    %           IF MULBY(RD.SOURCE) THEN                 % FUTURE FACILITIES
154565    % $*           LDT HDEV; AAT RCIVD; IOXT; STA RD     % STORE ALL BYTES READ
154565    %             RD.INFORMATION=:MULTIBYTES(RD.SOURCE, 5 - NOOFBYTES(RD.SOURCE))
154565    %             IF NOOFBYTES(RD.SOURCE)-1=:NOOFBYTES(RD.SOURCE) = 0 THEN
154565    %                FALS=:MULBY(RD.SOURCE); OKRET=: PB.RETFUNC % ALL BYTES READ
154565    %                IF  PB.POCTO >< NIL THEN
154565    %                   IF  PB.POCTO.RTRES >< 0 THEN $* COPY SB DD; COPY SX DB
154565    %                      RTACT; $* COPY SD DB
154565    %                   ELSE $* LDA NORES; CALL 9ERR(#99); 32460
154565    %                   FI
154565    %                FI
154565    %             FI
154565    %           ELSE $* LDA MULERR; CALL 9ERR(#99); 32460
154565    %           FI
154565   ENDDO:
154565   GO FAR BEGDO
154566   WHIL:
154566   "0"=:RC; AD SH -6; A:=OWNCPU SHR 4; AD SH 6; A SHR 6=:RC
154576   RC BONE RINTEN=:RC                                    % PREPARE NEXT INTERUPT
154601   T:= HDEV+ RCIVCW; A:= RC;* IOXT
154605   IF WQUEUE = NIL AND RECE = TRUE GO FAR WINT
154615
154615   %    "OUTPUT" PART OF THE OCTO-BUS DRIVER ***
154615   %         ALL TASK-CALLED FUNCTIONS - SEND WAITING MESSAGES
154615
154615   IF RECE = TRUE THEN                                   % TRUE ONLY IF SOMETHING
```

```
154621            FALS=:RECE; GO FAR TRLOOP                    % IN THE TRANSMIT QUEUE
154624    FI
154624    IF SLOTNO > MAXTASK THEN A:= ILSLOT; CALL 9ERR(#99); GO FAR OKFUNC FI
154634    A * OBOXSI + POSTBOX=:PB; DLEVL=: PB.LEVEL
154642    IF OFNC >< KICK GO FAR TSTIOW                % KICK_TASK
154646    * JMP FILL1; )FILL; FILL1=*; )KILL FILL1
154657
154657    %    PREPARE TRANSMIT WORD AND PUT MESSAGE INTO WQUEUE ***
154657
154657    T:= HDEV+ TRNSST;* IOXT; STA TS,B
154663    IF A:=PB.OMODE BIT WTMEI OR A:=TS BIT TSERROR THEN
154672       IF TS BIT TSERROR OR TS NBIT TFIEM THEN
154700    %     CHECK ALL ERRORS
154700          IF LASTSLOT >= 0 AND A < MAXTASK THEN
154705             IF TS NBIT TFIEM THEN
154710    %     PREV. TRANSMIT NOT COMPLETE
154710                LASTSLOT*14 + POSTBOX=:X; PRTRANS=:X.RETFUNC
154716    %     TRANSMIT INTERUPT ENABLE
154716                "0"=:TC; A BONE TINTEN=:TC
154722                T:= HDEV+ TRNSCW; A:= TC;* IOXT
154726             ELSE
154727                LASTSLOT*14 + POSTBOX=:X; PTRERR=:X.RETFUNC
154735             FI
154735             IF A:=PB.OMODE BIT WTMEI OR SLOTNO >< LASTSLOT THEN
154745                LASTSLOT*14 + POSTBOX=:X; TS=:X.TRNSTAT
154753             ELSE IF A:=PB.OMODE NBIT WTMEI THEN 0=:PB.TRNSTAT
154762             FI;FI
154762          FI
154762       FI
154762    ELSE
154763       0=:PB.TRNSTAT
154765    FI
154765
154765    %    CHECK PREVIOUS TRANSMIT
154765    IF PB.TRNSTAT >< 0 THEN
154770       A:=PB.OMODE BZERO WTMEI=:X.OMODE; PB.TERRCNT + 1 =: PB.TERRCNT
155001    ELSE
155002
155002    %                 PB APPEND WQUEUE:LINK
155002       IF A:=WQUEUE >< NIL THEN
155006          X:=PB=:D
155010          DO A:=A.LINK
155012          WHILE A >< 0
155013          OD
155014          A:=D=:X.LINK
155016       FI
155016       PB=:WQUEUE; NIL=: PB.LINK
155023    %     PREVIOUS TRANSMIT NOT FINISHED
155023       PRTRANS=:PB.RETFUNC
155026    %     MESSAGE IN TRANSMIT QUEUE
155026       A:=PB.OMODE BONE WTMEI=:X.OMODE
155032    %     SET UP TRANSMIT DATA
155032       T:=TDATA(SLOTNO); T BONE CONTROL=:TDATA(X)
155036       A:=DIRMINI; *BLDA 0 DA
155040       T:=TDATA(X); *BSTA 160 DT
155042       T=:TDATA(X)
155043       A:=CPU; AD SH -6; A:=TDATA(X)
155046       A SHR 2; AD SH 6; A SHR 10=:TDATA(X)
155052       "0"; *BLDA 0 DA
155054       T:=TDATA(X); *BSTA 70 DT
```

```
155056       T=:TDATA(X)
155057       "0"; AD SH -2; A:=TDATA(X) SHR 11
155063       AD SH 2; A SHR 5=:TDATA(X)
155066       A:=SLOTNO; AD SH -5; A:=TDATA(X) SHR 13
155072       AD SH 5; A=:TDATA(X)
155074    FI
155074    * JMP FILL1; )FILL; FILL1=*; )KILL FILL1
155076
155076    %    TRANSMIT MESSAGES FROM THE WQUEUE ***
155076
155076    TRLOOP :
155076    DO
155076    % READY FOR TRANSMIT
155076       WHILE X:=WQUEUE=:PB >< NIL AND TS BIT TFIEM
155106    %                          WQUEUE REMOVE WQUEUE:LINK
155106       X.LINK=:WQUEUE
155110       A:=PB.BOXN=:SLOTNO
155113    % CHECK PREVIOUS TRANSMIT
155113       IF PB.TRNSTAT >< 0 THEN
155116          PTRERR=: PB.RETFUNC
155121       ELSE
155122    %    TAKE NEXT
155122          A:=0=:TC
155124    %    ACTIVATE AND PREPARE INTERRUPT, CLEAR FIFO?
155124          A BONE ACTIVATE BONE TINTEN=:TC
155127    %    WRITE TRANSMIT CONTROL WORD
155127          T:= HDEV+ TRNSCW; A:= TC;* IOXT
155133          A:=TDATA(SLOTNO)=:TD
155136          SLOTNO=:LASTSLOT
155140    %    WRITE TRANSMIT DATA WORD
155140          T:= HDEV+ TRNSD;  A:= TD;* IOXT
155144          PB.KICKCNT + 1 =: X.KICKCNT
155150          TOTTRANS + 1 =:TOTTRANS
155153    %    WAIT 40 MICROS.
155153          FOR X:=-40 DO OD
155155          T:= HDEV+ TRNSST;* IOXT; STA TS,B
155161    %    READY FOR TRANSMIT
155161          IF TS BIT TFIEM OR A BIT TSERROR THEN
155166    %    OUT OF "WAIT-FOR-TRANSMIT" STATE
155166             A:=PB.OMODE BZERO WTMEI=:X.OMODE
155172    %    POSSIBLE ERRORS
155172             IF TS BIT TSERROR THEN
155175                                   TRERR=:PB.RETFUNC; TS=: PB.TRNSTAT
155203             ELSE OKRET=: PB.RETFUNC      % OK RETUR FUNCTION
155207             FI
155207          ELSE NOTTR=: PB.RETFUNC         % ERROR RETUR FUNCTION
155213          FI
155213       FI
155213    OD
155214    SLOTNO*14 + POSTBOX=:PB
155220    GO FAR RTASK
155221
155221    %  WAIT FOR KICK IN I/O WAIT
155221    TSTIOW:
155221    IF OFNC = IOW THEN
155225       IF  PB.POCTO >< NIL THEN
155232          IF  A:=A.RTRES >< 0 THEN X=:D; A=:T
155237             IF PB.OMODE NBIT REP THEN
155243                B:=:D; T=:X
155245                CALL WDATA; CALL SERET; D=:B
```

```
155250                A:=PB.OMODE BONE WKICK=:X.OMODE
155254             ELSE A:=PB.OMODE BZERO REP=:X.OMODE
155261                FALS; CALL RKTASK; CALL SERET
155264             FI
155264          FI
155264       ELSE NOTAS=: PB.RETFUNC
155270       FI; GO FAR OKFUNC;* )FILL
155277    FI
155277
155277    %  ACCEPT A KICK WHEN IT COMES
155277    IF OFNC = RTA THEN
155303       IF  PB.POCTO >< NIL THEN A:=TRENA;* BLDA 0 DA
155312          A:=PB.OMODE;* BSTA 160 DA
155315          A=:X.OMODE
155316          OKRET=: PB.RETFUNC % RESTART IF FUNCTION IS RTA
155321       ELSE NOTAS=: PB.RETFUNC
155325       FI; GO FAR RTASK
155326    FI
155326
155326    %  PREPARE WAIT FOR KICK
155326    IF OFNC = PKIC THEN
155332       IF  PB.POCTO = NIL THEN
155337          PPREG=: PB.PREG; OKRET=: PB.RETFUNC
155345       ELSE NORT=: PB.RETFUNC
155351       FI; GO FAR RTASK
155352    FI
155352
155352    %  SEND CONTROL PACKET ( FUNCTION NOT IMPLEMENTED!)                    --
155352    IF OFNC = SCPCK THEN SCPC=: PB.RETFUNC; GO FAR RTASK FI
155362
155362    %  READ LAST STATUS (CHECK LAST TRANSMIT)
155362    IF OFNC = RLSTA THEN
155366       IF  PB.OMODE BIT WTMEI THEN
155372          PRTRANS=: PB.RETFUNC          % PREVIOUS TRANSMIT NOT FINISHED
155375       ELSE IF  PB.TRNSTAT >< 0 THEN  % ERROR IN LAST TRANSMIT
155401          PTRERR=: PB.RETFUNC
155404       ELSE OKRET=: PB.RETFUNC
155410       FI;FI
155410       T:= HDEV+ TRNSST;* IOXT          % READ TRANSMIT STATUS
155413       A=:TS; T:= HDEV+ RCIVST;* IOXT; STA RS,B % READ RECEIVE STATUS
155420       IF  X:=PB.POCTO = NIL THEN       % DIRECT TASK?
155425    WREG :
155425          PB.LEVEL + INTRW + 1=:T
155432    %    IRW LVXXB DD LAST TRANSMIT STATUS TO D-REG
155432          A:=PB.TRNSTAT;* EXR ST
155435    %    ACTUAL RECEIVE STATUS TO T-REG
155435          T+ 6; A:= RS;* EXR ST
155440    %    ACTUAL TRANSMIT STATUS TO X-REG
155440          T+ 1; A:= TS;* EXR ST
155443       ELSE IF CURPROG = X.RTRES THEN 10=: PB.LEVEL; GO WREG
155454          * )FILL
155457       ELSE
155460          A:=TS=:D:=PB.TRNSTAT; T=:X;* LDX 16,X; LDT RS,B
155467    %  LAST USED TRANSMIT STATUS FOR THIS SLOT
155467          * STA DDREG,X
155470    %  ACTUAL HARDWARE RECEIVE STATUS
155470          * STT DTREG,X; COPY SD DA
155472    %  ACTUAL HARDWARE TRANSMIT STATUS
155472          * STA DXREG,X
155473       FI;FI
```

```
155473      0=: PB.TRNSTAT; GO FAR RTASK
155476   FI
155476
155476   %  WHO AM I
155476   IF OFNC = WHOAMI THEN
155502      T:= HDEV+ RCIVST;* IOXT        % READ RECEIVE STATUS
155505      A /\ 37400 SH -10=:RS
155510   %   TEST IF DIRECT TASK?
155510      IF  X:=PB.POCTO = NIL THEN
155515   %   SET UP INSTRUCTION :  IRW XX DT
155515   WCPU :
155515         PB.LEVEL + INTRW + 6=:T
155522   %   ACTUAL CPU NUMBER TO T-REG
155522         A:=RS;* EXR ST
155524   %   IT IS AN RT-PROGRAM
155524      ELSE IF CURPROG = X.RTRES  THEN 10=: PB.LEVEL; GO WCPU
155535      ELSE
155536         T=:X;* LDX 16,X; LDA RS,B; STA DTREG,X
155542      FI;FI
155542      OKRET=: PB.RETFUNC; GO RTASK
155546   FI
155546   WFU=: PB.RETFUNC
155551
155551   RTASK:
155551   FALS; CALL RKTASK; CALL 5ERET
155554
155554   %    READ TRANSMIT STATUS
155554   OKFUNC:
155554   T:= HDEV+ TRNSST;* IOXT; STA TS,B
155560   IF TS BIT TFIEM AND WQUEUE >< NIL THEN GO FAR TRLOOP FI
155570
155570   %   READ RECEIVE STATUS
155570   WINT:
155570   T:= HDEV+ RCIVST;* IOXT; STA RS,B
155574   IF RS NBIT FIFORFTR THEN FALS=:RECE
155601
155601   %   CHECK FOR MORE RECEIVE
155601      CALL ID13
155602   FI
155602
155602   %  START ON NEXT INTERUPT
155602   TRUE=:RECE
155604   GO FAR BEGDO
155605   %                                                    OCTOBUS   %
155605   * )FILL
155616
155616   %=====================================================================%
155616   %       S T A R C                                                     %
155616   %=====================================================================%
155616   % ONLY CALLED FIRST TIME INTERUPT OCCURES
155616   % NEXT TIME THE DRIVER WILL "WAIT" IN ID13!
155616
155616
155616   STARC:
155616   TRUE=:RECE
155620   CALL OCTOBUS; * WAIT; JMP *
155623   %                                                      STARC   %
155623
155623   %=====================================================================%
155623   %       R E S T _ T A S K                                             %
```

```
155623   %=================================================================%
155623
155623   % RESTART RT-PROGRAM IF A-REG >< 0 (TRUE!)
155623   % CODE TO RESTART TASK AFTER EXECUTING OCTODRIV MONCALL
155623   % OR AFTER "KICKED" FROM ANOTHER TASK
155623   % RETFUNC  = A-REG; RETURN FUNCTION >< 0 MEANS ERROR (SEE ERROR LIST)
155623   % TRNSTAT  = OCTOBUS TRANSMIT STATUS IF PREVIOUS TRANSMIT WENT WRONG
155623   %             RECEIVED BY CALLING TASK ON NEXT OCTOBUS CALL
155623   % PB       = ACTUAL POSTBOKS
155623
155623
155623   RKTASK:
155623   %  RESTART KICKED TASK
155623   %  SAVE RESTART FLAG, TRUE IF RESTART - FALS IF NOT RESTART
155623   %
155623   A=:D
155624   %  SAVE RETURN ADDRESS
155624   A:=L=: RETUR
155626   IF  PB.POCTO >< NIL THEN
155633      IF  A.RTRES >< 0 THEN
155636         IF D = FALS GO NOTREST
155641   %  START RT-PROG
155641         B=:D; X=:B; A=:X
155644         CALL RTACT; D=:B
155646   %  DON'T START RT-PROG!!!
155646   NOTREST :
155646         * IOF
155647         X:=PB.POCTO
155651         IF CURPROG =  X.RTRES THEN
155655            A:=PB.RETFUNC;* IRW ALEVB DA; COPY SA DD
155661            * IRR ALEVB DP; SKP IF DD UEQ 0; AAA 1; IRW ALEVB DP
155665            * ION
155666         ELSE
155667            A:=PB.RETFUNC; T=:X;* LDX 16,X     % D-REG ADDRESS OF REG. BLCK
155673            * STA DAREG,X; SKP IF DA UEQ 0; MIN DPREG,X; JMP *1
155677            * ION
155700         FI
155700         OKRET=: PB.RETFUNC
155703      ELSE A:= NORES; CALL 9ERR(#99)
155707      FI
155707   ELSE
155710   %  CHECK LEVEL OF DIRECT TASK
155710            A:=PB.LEVEL=:BSTDA=:D
155714            T:= BSETO; T+A
155716   %  ENABLE DIRECT TASK LEVEL FOR RETURN
155716            "0";* EXR ST; MST PID                  % BSET ONE XX DA; MST PID
155721
155721   %   POSSIBLE SKIP RETURN OF DIRECT TASK - NOT USED
155721   %        $* LDT INTRR; RADD SD DT; AAT 2; EXR ST; STA UTILL % IRR LVXXB DP
155721
155721   %   IRW LVXXB DA RETURN VALUE
155721            A:=PB.RETFUNC; T:= INTRW; T + D + 5;* EXR ST
155727
155727   %        IF  PB.RETFUNC = 0 THEN
155727   %           $* AAT -3; LDA UTILL; AAA 1; EXR ST
155727   %        FI
155727
155727            OKRET=: PB.RETFUNC
155732   FI
155732   A:= RETUR=: L; EXITA
```

```
155735
155735    % DUMMY ERROR ROUTINE
155735    SERET: EXIT
155736
155736    %=======================================================================%
155736    %=!     S Y S E R R                                                     %
155736    %=======================================================================%
155736    SYSERR: * IOF; WAIT; JMP *
155741    RBUS
155747    % ENDMODULE
155747    *"
"155747
155747
155747    @DEV 1
155747    @DEV (S-S-J)MRES-SSCOM
155747
```

```
155747
155747   %%%%%%%%%%%%%%%%%%%%%%% M R E S - S S C O M %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
155747   *"BDTEX
155747
155747
155747   *"CASY
155747
155747   @DEV 1
155747   @DEV (S-S-J)COS-TAD-POF-CODE
155747
155747   %%%%%%%%%%%%%%%%%%%%%%% COS-TAD-POF-CODE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
155747
155747   *"BADAD
155747    %=============================================================================
155747   %       T * A * D   P * A * G * I * N * G   O * F * F   P * A * R * T
155747   %=============================================================================
155747
155747   %=============================================================================
155747   %       T A D   B U F F E R   O P E R A T I O N   R O U T I N E S
155747   %=============================================================================
155747
155747   % XX.XX       D R X A C C
155747   %
155747   % SUBROUTINE TO PERFORM DIRECT ACCESS TO XMSG BUFFERS
155747   % NOTE! THE ROUTINE TURNS THE INTERRUPT OFF AND RETURNS IN IOF
155747   SUBR DRXACC
155747   INTEGER POINTER LREG
155750   INTEGER FUNC=?
155750   DISP 0; INTEGER ACCFUN; PSID
155750   DRXACC: IF X=0 THEN A:=XEIBP; EXITA; FI          % INCORRECT BUFFID
155754           X:=:L; A=:D; X.ACCFUN; *IOF
155760           A=:FUNC; X=:"LREG"; L=:X; D=:A
155764           CALL DRXMSG; INTEGER FUNC
155766           GO RETU; MIN "LREG"
155770   RETU:   MIN "LREG"; GO LREG
155772   RBUS
155773
155773   %=============================================================================
155773   %       W R M H E A D
155773   %
155773   % ROUTINE TO WRITE MESSAGE HEADER ON EVEN BYTE
155773   %       ENTRY:    A-REG - MESSAGE HEADER
155773   %                 B-REG - TAD OUTPUT DATAFIELD
155773   %       RETURN:   EXIT
155773   %
155773   SUBR WRMHEAD
155773   WRMHEAD: A=:D; T:=TDTAFI; X:=TDTALA
155776           IF TDBTPT BIT "0" THEN                  % ODD START, CREATE PAD BYTE
156001              A SHZ -1; X+A; *LDATX
156004              A/\177400; *STATX
156006              MIN TDBTPT; REMSIZ-1=:REMSIZ
156012           FI
156012           X:=TDTALA
156013           TDBTPT SHZ -1; X+A; D=:A; *STATX             % WRITE IN BUFFER
156020           TDBTPT+2=:TDBTPT; REMSIZ-2=:REMSIZ
156026           EXIT
156027   RBUS
156030
156030   %=============================================================================
156030   % XX.XX       C R E M E S
```

```
156030    %
156030    % ROUTINES TO CREATE MESSAGE-HEADERS IN XMSG BUFFER  (TAD OUTPUT)
156030    %
156030    % ENTRY:      A-REG - MESSAGE TYPE
156030    %             T-REG - REQUIRED I-FIELD SIZE
156030    %             B-REG - TAD OUTPUT DATAFIELD
156030    %
156030    % RETURN:     SKIP-RETURN:  OK      TAD DATAFIELD BUFFER VARIABLES
156030    %                                   REMSIZ,CURMES,NOBDIS ARE UPDATED
156030    %
156030    %             NOSKIP:       ERROR   A-REG:   1  NO OUTPUT BUFFER PRESENT
156030    %                                   A-REG:   2  OUTPUT-BUFFER FULL
156030    %                                   A-REG: <0   XMSG ERROR CODE
156030    %
156030    SUBR CREMES
156030    DISP -2; INTEGER AREG,TREG; PSID
156030    CREMES: L=:D; CALL SETSTACK(AREG)
156033            CALL STAB(AREG); CALL STTB(TREG)
156037            IF BUFFID=0 THEN CALL GETPOOL; GO NOPOL FI          % NO BUFFER
156043            CALL LDTB(TREG)                                     % GET REQUIRED IFIELD SIZE
156045            IF TDBTPT BIT "0" THEN T+3 ELSE T+2 FI              % ODD OR EVEN START IN BUFFER
156053            IF REMSIZ<T THEN A:=2; GO T1BEXI FI                 % NOT ENOUGH SPACE
156060            CALL LDAB(AREG); A SHZ 10; CALL WRMHEAD             % WRITE MESSAGE HEADER
156064            TDBTPT-1=:NOBDIS; CALL LDAB(AREG); A=:CURMES        % UPDATE MESSAGE INFO
156072            GO T2BEXI
156073    NOPOL:  A:=1; GO T1BEXI
156075    RBUS
156106
156106    %=====================================================================
156106    %             C R H E E V
156106    %
156106    % ROUTINE LIKE CREMES BUT BYTE COUNTER IS ALSO WRITTEN
156106    % USED WHEN MESSAGE IS COMPLETED BY CALLING ROUTINE
156106    % MESSAGE HEADER IS CREATED ON EVEN BYTE
156106    %
156106    % ENTRY:      A-REG - MESSAGE TYPE
156106    %             T-REG - REQUIRED I-FIELD SIZE
156106    %             B-REG - TAD OUTPUT DATAFIELD
156106    %
156106    % RETURN:     SKIP-RETURN:  OK      TAD DATAFIELD BUFFER VARIABLES
156106    %                                   REMSIZ,CURMES,NOBDIS ARE UPDATED
156106    %
156106    %             NOSKIP:       ERROR   A-REG:   1  NO OUTPUT BUFFER PRESENT
156106    %                                   A-REG:   2  OUTPUT-BUFFER FULL
156106    %
156106    SUBR CRHEEV
156106    DISP -2; INTEGER AREG,TREG; PSID
156106    CRHEEV: L=:D; CALL SETSTACK(AREG)
156111            CALL STAB(AREG); CALL STTB(TREG)
156115            IF BUFFID=0 THEN CALL GETPOOL; GO NOPOL FI          % NO BUFFER
156121            CALL LDTB(TREG)                                     % GET REQUIRED IFIELD SIZE
156123            IF TDBTPT BIT "0" THEN T+3 ELSE T+2 FI              % ODD OR EVEN START IN BUFFER
156131            IF REMSIZ<T THEN A:=2; GO T1BEXI FI                 % NOT ENOUGH SPACE
156136            CALL LDAB(AREG); A SHZ 10; CALL LDTB(TREG)
156143            A+T; CALL WRMHEAD                                   % WRITE MESSAGE HEADER
156145            TDBTPT-1=:NOBDIS; CALL LDAB(AREG); A=:CURMES        % UPDATE MESSAGE INFO
156153            GO T2BEXI
156154    NOPOL:  A:=1; GO T1BEXI
156156    RBUS
156167
```

```
156167   %=====================================================================
156167   %              C R H E O D
156167   %
156167   % ROUTINE LIKE CREMES BUT BYTE COUNTER IS ALSO WRITTEN
156167   % USED WHEN MESSAGE IS COMPLETED BY CALLING ROUTINE
156167   % MESSAGE HEADER IS CREATED ON ODD BYTE
156167   %
156167   % ENTRY:       A-REG - MESSAGE TYPE
156167   %              T-REG - REQUIRED I-FIELD SIZE
156167   %              B-REG - TAD OUTPUT DATAFIELD
156167   %
156167   % RETURN:      SKIP-RETURN:  OK      TAD DATAFIELD BUFFER VARIABLES
156167   %                                    REMSIZ,CURMES,NOBDIS ARE UPDATED
156167   %
156167   %              NOSKIP:       ERROR   A-REG:  1  NO OUTPUT BUFFER PRESENT
156167   %                                    A-REG:  2  OUTPUT-BUFFER FULL
156167   %
156167   SUBR CRHEOD
156167   DISP -2; INTEGER AREG,TREG; PSID
156167   CRHEOD: L=:D; CALL SETSTACK(AREG)
156172           CALL STAB(AREG); CALL STTB(TREG)
156176           IF BUFFID=0 THEN CALL GETPOOL; GO NOPOL FI          % NO BUFFER
156202           CALL LDTB(TREG)                                     % GET REQUIRED IFIELD SIZE
156204           IF TDBTPT NBIT "0" THEN T+3 ELSE T+2 FI             % ODD OR EVEN START IN BUFFER
156212           IF REMSIZ<T THEN A:=2; GO T1BEXI FI                 % NOT ENOUGH SPACE
156217           IF TDBTPT NBIT "0" THEN A:=0; CALL STORBYT FI       % EVEN START, CREATE PAD BYTE
156224           CALL LDAB(AREG); CALL STORBYT                       % WRITE MESSAGE TYPE
156227           CALL LDAB(TREG); CALL STORBYT                       % WRITE BYTE COUNTER
156232           TDBTPT-1=:NOBDIS; CALL LDAB(AREG); A=:CURMES        % UPDATE MESSAGE INFO
156240           GO T2BEXI
156241   NOPOL:  A:=1; GO T1BEXI
156243   RBUS
156254
156254   %=====================================================================
156254   % XX.XX        G E T M E S
156254   %
156254   % ROUTINE TO FIND NEXT MESSAGE HEADER IN XMSG BUFFER  (TAD INPUT)
156254   % ENTRY:       B-REG - TAD INPUT DATAFIELD
156254   %
156254   % RETURN:      SKIP-RETURN:  OK      A-REG:  MESSAGE-TYPE
156254   %                                    T-REG:  NUMBER OF BYTES IN I-FIELD
156254   %                                    TAD DATAFIELD BUFFER VARIABLES
156254   %                                    REMSIZ,TDBTPT,CURMES,REMBYT ARE UPDATED
156254   %
156254   %              NOSKIP:       ERROR   A-REG:  1  NO INPUT BUFFER
156254   %                                    A-REG:  2  BUFFER IS EMPTY
156254   %                                    A-REG:  3  INCONSISTENCY, MESSAGE BIGGER THAN BUFFER
156254   SUBR GETMES
156254   GETMES: IF BUFFID=0 THEN A:=1; EXIT FI                      % NO BUFFER
156260   TSTSP:   IF REMSIZ<2 THEN T=:A; EXIT FI
156266            T:=TDTAFI; X:=TDTALA; TDBTPT=:D SHZ -1
156273            X+A; *LDATX
156275            MIN TDBTPT
156276            IF D BIT "0" THEN                                 % ODD BYTE
156300               IF A/\377=0 THEN REMSIZ-1=:REMSIZ; GO TSTSP FI % FIRST BYTE FOUND
156306               A=:CURMES; X+1; *LDATX
156311               A SHZ -10; MIN TDBTPT; GO RCHK                 % RETURN AND CHECK
156314            FI
156314            A=:T SHZ -10
156316            IF A=0 THEN
```

```
156317              REMSIZ-1=:REMSIZ; GO TSTSP                          % PAD BYTE
156323            FI
156323            A=:CURMES:=T/\377; MIN TDBTPT
156327    RCHK:   IF A>REMSIZ-2 THEN A:=3; EXIT FI                      % INCONSISTENT MESSAGE
156335            A=:T+1-=:REMBYT; X:=REMSIZ-2=:REMSIZ
156344            A:=CURMES; EXITA                                      % MESSAGE FOUND
156346    RBUS
156347
156347    %=============================================================================
156347    %               U P D M B C
156347    %
156347    % ROUTINE TO UPDATE MESSAGE BYTE COUNT
156347    % ENTRY:          A-REG - NUMBER OF BYTES
156347    %                 B-REG - TAD OUTPUT DATAFIELD
156347    % SKIP RETURN: OK
156347    % RETURN:         NOT ENOUGH SPACE
156347    SUBR UPDMBC
156347    UPDMBC: A=:D; T:=TDTAFI; X:=TDTALA
156352            IF NOBDIS BIT "0" THEN                                % BYTE COUNT ON ODD BYTE
156355               A SHZ -1; X+A; *LDATX                              % READ BYTE COUNT
156360               A=:T; A/\177400; A:=:T; A/\377
156364               A+D; T=:D; IF A>=400 THEN EXIT FI                  % MESSAGE FULL?
156372               A\/D
156373            ELSE                                                  % BYTE COUNT ON EVEN BYTE
156374               A SHZ -1; X+A; *LDATX                              % READ BYTE COUNT
156377               A=:T; A/\377; A:=:T; A SHZ -10
156403               A+D; T=:D; IF A>=400 THEN EXIT FI                  % MESSAGE FULL?
156411               A SHZ 10; A\/D
156413            FI
156413            T:=TDTAFI; *STATX                                     % WRITE BACK BYTE COUNT
156415            EXITA
156416    RBUS
156421
156421    %=============================================================================
156421    %               G E T M B C
156421    %
156421    % ROUTINE TO FIND NUMBER OF BYTES IN CURRENT OUTPUT MESSAGE
156421    %
156421    % ENTRY:          B-REG - TAD OUTPUT DATAFIELD
156421    % RETURN:         A-REG - BYTE COUNT
156421    SUBR GETMBC
156421    GETMBC: IF NOBDIS=0 THEN A:=400; EXIT FI
156425            A=:D SHZ -1; T:=TDTAFI; X:=TDTALA; X+A; *LDATX
156433            IF D BIT "0" THEN A/\377 ELSE A SHZ -10 FI
156440            EXIT
156441    RBUS
156443
156443    %=============================================================================
156443    % XX.XX         B Y T P U T
156443    %
156443    % ROUTINE TO PUT A BYTE IN XMSG BUFFER  (TAD OUTPUT)
156443    %
156443    % ENTRY:          A-REG - BYTE
156443    %                 B-REG - TAD OUTPUT DATAFIELD
156443    %
156443    % RETURN:         SKIP-RETURN:  OK      TAD DATAFIELD BUFFER VARIABLES
156443    %                                       REMSIZ AND TDBTPT IS UPDATED
156443    %
156443    %               NOSKIP:       ERROR   A-REG:  1  NO OUTPUT BUFFER PRESENT
156443    %                                       A-REG:  2  OUTPUT BUFFER FULL
```

```
156443    %                                   A-REG:  3  MESSAGE FULL
156443    SUBR BYTPUT
156443    DISP -1; INTEGER AREG; PSID
156443    BYTPUT: L=:D; CALL SETSTACK(AREG); A/\377; CALL STAB(AREG)
156451            IF BUFFID=0 THEN A:=1; GO T1BEXI FI          % OUTPUT BUFFER PRESENT?
156455            IF REMSIZ<=0 THEN A:=2; GO T1BEXI FI         % BUFFER FULL?
156462            A:=1; CALL UPDMBC; GO MSFUL                  % UPDATE MESSAGE BYTE COUNT
156465            T:=TDTAFI; X:=TDTALA; TDBTPT=:D SHZ -1; X+A
156473            IF D BIT "0" THEN                            % DESTINATION ODD BYTE
156475               *LDATX
156476               A/\177400; CALL LDTB(AREG); A\/T
156502            ELSE                                         % DESTINATION EVEN BYTE
156503               CALL LDAB(AREG); A SHZ 10
156506            FI
156506            T:=TDTAFI; *STATX                          % PUT BYTE IN BUFFER
156510            REMSIZ-1=:REMSIZ; MIN TDBTPT; GO T2BEXI    % UPDATE BUFFER COUNTERS AND RETURN
156515    MSFUL: A:=3; GO T1BEXI                             % MESSAGE IS FULL
156517    RBUS
156530
156530    %-------------------------------------------------------------------
156530    % XX.XX       W O R D P U T
156530    %
156530    % ROUTINE TO PUT A WORD IN XMSG BUFFER  (TAD OUTPUT)
156530    %
156530    % ENTRY:      A-REG - WORD
156530    %             B-REG - TAD OUTPUT DATAFIELD
156530    %
156530    % RETURN:     SKIP-RETURN:  OK      TAD DATAFIELD BUFFER VARIABLES
156530    %                                   REMSIZ AND TDBTPT IS UPDATED
156530    %
156530    %             NOSKIP:       ERROR   A-REG:  1  NO OUTPUT BUFFER PRESENT
156530    %                                   A-REG:  2  OUTPUT BUFFER FULL
156530    %                                   A-REG:  3  MESSAGE FULL
156530    SUBR WORDPUT
156530    DISP -1; INTEGER AREG; PSID
156530    WORDPUT: L=:D; CALL SETSTACK(AREG); CALL STAB(AREG)
156535            IF BUFFID=0 THEN A:=1; GO T1BEXI FI          % OUTPUT BUFFER PRESENT?
156541            IF REMSIZ<=1 THEN A:=2; GO T1BEXI FI         % BUFFER FULL?
156547            A:=2; CALL UPDMBC; GO MSFUL                  % UPDATE MESSAGE BYTE COUNT
156552            T:=TDTAFI; X:=TDTALA; TDBTPT=:D SHZ -1; X+A
156560            IF D BIT "0" THEN                            % ODD START IN BUFFER
156562               *LDATX
156563               A/\177400; CALL LDTB(AREG); T:=:A
156567               A SHZ -10; A\/T; T:=TDTAFI; *STATX        % WRITE FIRST BYTE
156573               CALL LDAB(AREG); A SHZ 10; X+1; *STATX    % WRITE SECOND BYTE
156600            ELSE                                         % EVEN START IN BUFFER
156601               CALL LDAB(AREG); *STATX                   % WRITE WORD IN BUFFER
156604            FI
156604            REMSIZ-2=:REMSIZ; MIN TDBTPT; MIN TDBTPT  % UPDATE BUFFER COUNTERS
156611            GO T2BEXI                                    % RETURN
156612    MSFUL: A:=3; GO T1BEXI                               % MESSAGE IS FULL
156614    RBUS
156624
156624    %-------------------------------------------------------------------
156624    % XX.XX       B Y T G E T
156624    %
156624    % ROUTINE TO GET A BYTE IN XMSG BUFFER  (TAD INPUT)
156624    % MUST BE CALLED WITH PAGING INTERUPT OFF!
156624    % ENTRY:      B-REG - TAD INPUT DATAFIELD
156624    %
```

```
156624   % RETURN:     SKIP-RETURN:  OK      A-REG:  BYTE
156624   %                                   TAD DATAFIELD BUFFER VARIABLES
156624   %                                   REMSIZ,REMBYT ARE UPDATED
156624   %
156624   %             NOSKIP:       ERROR   A-REG:  0  MESSAGE EMPTY
156624   %                                   A-REG:  1  NO INPUT BUFFER PRESENT
156624   %                                   A-REG:  2  BUFFER IS EMPTY
156624   SUBR BYTGET
156624   BYTGET:IF BUFFID=0  THEN A:=1; EXIT FI                  % NO INPUT BUFFER
156630          IF REMSIZ<=0 THEN A:=2; EXIT FI                  % INPUT BUFFER EMPTY
156635          IF REMBYT=0  THEN EXIT FI                        % MESSAGE EMPTY
156640          MIN REMBYT; GO CONT
156642          A:=0=:CURMES; EXIT                               % MESSAGE EMPTY
156645   CONT:  T:=TDTAFI; X:=TDTALA
156647          TDBTPT=:D SHZ -1; X+A; *LDATX                    % GET WORD WITH BYTE
156654          IF D BIT "0" THEN A/\377 ELSE A SHZ -10 FI       % GET WANTED BYTE
156661          T:=REMSIZ-1=:REMSIZ; MIN TDBTPT
156665          EXITA
156666   RBUS
```

```
156667
156667
156667   %=====================================================================
156667   %       L O A D B Y T
156667   %
156667   % ROUTINE TO LOAD A BYTE FROM THE XMSG BUFFER ACCORDING TO TDBTPT
156667   % TDBTPT IS INCREMEMENTED
156667   % ENTRY:       B-REG  - TAD-DATAFIELD  INPUT OR OUTPUT
156667   % RETURN:      A-REG  - BYTE   (OTHER REGISTERS IS NOT MODIFIED)
156667   SUBR LOADBYT
156667   INTEGER XREG,TREG,DREG
156672   LOADBYT: *IOF
156673           T=:TREG:=D=:DREG; X=:XREG
156677           T:=TDTAFI; X:=TDTALA; TDBTPT=:D SHZ -1; X+A; *LDATX
156706           MIN TDBTPT; IF D BIT "0" THEN A/\377 ELSE A SHZ -10 FI
156714           T:=DREG=:D:=TREG; X:=XREG; *ION; EXIT
156722   RBUS
156723
156723   %=====================================================================
156723   %       S T O R B Y T
156723   %
156723   % ROUTINE TO STORE A BYTE IN THE XMSG BUFFER ACCORDING TO TDBTPT
156723   % TDBTPT AND REMSIZ IS UPDATED
156723   % ENTRY:       B-REG  - TAD-DATAFIELD  INPUT OR OUTPUT
156723   %              A-REG  - BYTE
156723   % RETURN:      NO REGISTERS MODIFIED
156723   SUBR STORBYT
156723   INTEGER TREG,AREG,DREG,XREG; TRIPLE TADREG=TREG
156727   STORBYT: *IOF
156730           TAD=:TADREG; X=:XREG
156732           T:=TDTAFI; X:=TDTALA; TDBTPT=:D SHZ -1; X+A
156740           IF D BIT "0" THEN
156742              *LDATX
156743              A/\177400=:T; AREG\/T; T:=TDTAFI; *STATX
156751           ELSE
156752              AREG SHZ 10; *STATX
156755           FI
156755           MIN TDBTPT; REMSIZ-1=:REMSIZ
156761           TAD:=TADREG; X:=XREG; *ION; EXIT
156765   RBUS
156766
156766   %=====================================================================
156766   %       T A D   I O T R A N S   R O U T I N E S
156766   %=====================================================================
156766
156766   % XX.XX       B D P U T
156766   %
156766   % OUTPUT IOTRANS ROUTINE FOR TAD.
156766   % CALLED WITH PAGING INTERUPT OFF
156766   % ENTRY:          A-REG - BYTE
156766   %                 B-REG - TAD OUTPUT DATAFIELD
156766   % SKIP RETURN:    OK
156766   % RETURN:         ERROR-CODE IN DERROR
156766   SUBR BDPUT
156766   BDPUT: A=:LAST:=L=:"LRSA"; X=:XRSA; 0=:TMR
156773           CALL MLVLOC; CALL BRSTACK
156775           IF DFOPP.PORTNO=0 THEN MIN "LRSA"; GO RETU; FI       % NOT CONNECTED
157002           IF 7BDAT><CURMES THEN
157006   CRMES:     T:=3; CALL CREMES; GO ERR
157011           FI
```

```
157011          LAST; CALL BYTPUT; GO ERR
157014          MIN "LRSA" GO RETU                                      % SKIP RETURN OK
157016
157016  ERR:    IF A<0 GO CNXE                                          % ERROR FROM XMSG
157017          IF A=1 GO RETU                                          % WAIT FOR BUFFER
157022          IF A=2 THEN                                             % BUFFER FULL
157025             CALL SNDBUF; GO CNXE
157027             IF BUFFID><0 THEN 7BDAT; GO CRMES FI
157033             GO RETU
157034          FI
157034          IF A=3 THEN 7BDAT; GO CRMES; FI                         % MESSAGE FULL
157041          GO RETU
157042  CNXE:   CALL CNVERR                                             % CONVERT ERROR-CODE
157043  EROUT:  A=:DERROR; X:=XRSA; CALL MLVULOC; GO LRSA
157047  RETU:   TTMR=:TMR; X:=XRSA; LAST; CALL MLVULOC; GO LRSA
157055  RBUS
157064
157064  %=======================================================================
157064  % XX.XX       B D G E T
157064  %
157064  % INPUT IOTRANS ROUTINE FOR TAD.
157064  % CALLED WITH PAGING INTERUPT OFF
157064  % ENTRY:          B-REG - TAD INPUT DATAFIELD
157064  % SKIP RETURN:    OK
157064  %                 A-REG - BYTE
157064  % RETURN:         ERROR-CODE IN DERROR
157064  SUBR BDGET
157064  BDGET:  A:=L=:"LRSA"; X=:XRSA; CALL MLVLOC
157070          CALL BRSTACK; CALL IEDCHK; GO EROUT         % INPUT WHILE DELAYED ESCAPE
157073          IF PORTNO=0 THEN A:=TER02; GO EROUT; FI     % NOT CONNECTED
157077  BYGET:  IF CURMES=7BDAT THEN
157103             CALL BYTGET; GO ERR; A=:LAST; MIN "LRSA"; GO RETU
157110          FI
157110  NXMES:  CALL GETMES; GO ERR
157112          IF A=X;=7BDAT THEN
157115             IF T=0 THEN GO NXMES ELSE GO BYGET FI
157121          ELSE
157122             CALL CTRMES; GO CNXE; GO BYGET
157125          FI
157125          GO RETU
157126
157126  ERR:    IF A<0 GO CNXE                                          % XMSG ERROR CODE
157127          IF A=0 GO NXMES                                         % DATA MESSAGE EMPTY
157130          IF A=1 OR A=2 THEN CALL SNDRFI; GO CNXE; GO RETU;FI     % BUFFER EMPTY, SEND RFI
157141          IF A=3 THEN                                             % INCONSISTENT MESSAGE
157144             CALL SNDREJ; GO CNXE; A:=TER01; GO EROUT             % MESSAGE REJECTED
157150          FI
157150          GO RETU
157151  CNXE:   CALL CNVERR
157152  EROUT:  A=:DERROR
157153  RETU:   X:=XRSA; CALL MLVULOC; GO LRSA
157156  RBUS
157172
157172  %=======================================================================
157172  % XX.XX       B M 8 O U T   B B 8 O U T   B D T C H
157172  %
157172  % BM8OUT (MON 22):
157172  %    MONITOR CALL TO WRITE UP TO 8 BYTES ON A TAD (0 BYTE TERMINATES)
157172  % BB8OUT (MON 24):
157172  %    MONITOR CALL TO WRITE 8 BYTES ON A TAD
```

```
157172    %
157172    % CALLED WITH PAGING INTERUPT OFF WITH B = OUTPUT-DATAFIELD
157172
157172    SUBR BM8OUT,BB8OUT,BDTCH
157172
157172    % ROUTINE TO CHECK FOR TERMINATION IN DATA
157172    BDTCH:   A=:D
157173             IF X:=XRSA=0 THEN
157176               A SHZ -10; IF A=0 THEN EXIT FI       % FIRST BYTE EMPTY
157201               D=:A; A/\377
157203               IF A=0 THEN A:=D SHZ -10; EXITA FI   % LAST BYTE EMPTY
157207             FI
157207             D=:A; L+1; EXITA
157212
157212    BM8OUT:  0=:XRSA; GO FELLS
157214    BB8OUT:  1=:XRSA
157216    FELLS:  IF DFOPP.PORTNO=0 GO MOURET                 % NOT CONNECTED
157222            0=:TMR; CALL BRSTACK                        % RESET TAD STACK
157224            IF 7BDAT><CURMES THEN                       % CURRENT NOT DATA-MESS
157230    CRMES:     T:=10; CALL CREMES; GO ERR               % RESERVE 10 BYTES
157233            ELSE
157234               IF REMSIZ<12 THEN A:=2; GO ERR FI        % NOT ENOUGH SPACE IN BUFFER
157242               CALL GETMBC                              % GET BYTE COUNT IN CURMES
157243               IF A+10>377 THEN 7BDAT; GO CRMES FI      % NOT ENOUGH SPACE IN MESSAGE
157251            FI
157251
157251    % ENOUGH SPACE IN BUFFER
157251
157251             *IRR ALEVB DA
157252             CALL BDTCH; GO OKRET; GO LASBY; CALL WORDPUT; GO ERR; *IRR ALEVB DD
157260             CALL BDTCH; GO OKRET; GO LASBY; CALL WORDPUT; GO ERR; *IRR ALEVB DL
157266             CALL BDTCH; GO OKRET; GO LASBY; CALL WORDPUT; GO ERR; *IRR ALEVB DX
157274             CALL BDTCH; GO OKRET; GO LASBY; CALL WORDPUT; GO ERR
157301
157301    % NORMAL RETURN, BYTES STORED IN BUFFER
157301    OKRET:  TTMR=:TMR; GO MOURET
157304
157304    % ONE BYTE LEFT TO WRITE IN BUFFER
157304    LASBY:  CALL BYTPUT; GO ERR; GO OKRET
157307
157307    % ERRORS FROM CREMES
157307    ERR:     IF A=1 GO RETU                             % WAIT FOR BUFFER
157312             IF A=2 THEN                                % BUFFER FULL
157315                CALL SNDBUF; GO CNXE
157317                IF BUFFID><0 THEN 7BDAT; GO CRMES FI    % NEW BUFFER
157323                GO RETU                                 % WAIT FOR BUFFER
157324             FI
157324             GO RETU
157325
157325    CNXE:   CALL CNVERR; A=:DERROR
157327    RETU:   TTMR=:TMR; GO WDX
157332
157332    RBUS
157345
157345    %=========================================================================
157345    % XX.XX       B B 4 I N W
157345    %
157345    % MON 63: MONITOR CALL ALWAYS RETURNS 8 BYTES FROM A TAD
157345    % CALLED WITH PAGING INTERUPT OFF WITH B = INPUT DATAFIELD
157345
```

```
157345    SUBR BB4INW
157345    BB4INW: CALL BRSTACK; CALL IEDCHK; GO EROUT        % INPUT WHILE DELAYED ESCAPE
157350            IF PORTNO=0 THEN A:=TER02; GO EROUT FI     % NOT CONNECTED
157354    POSDA:  IF CURMES><7BDAT THEN                      % NOT DATA MESSAGE
157360    NXMES:     CALL GETMES; GO ERR                     % GET NEW MESSAGE
157362               IF A><X:=7BDAT THEN                     % CONTROLL MESSAGE
157365                  CALL CTRMES; GO CNXE; GO POSDA       % PERFORM CONTROLL
157370               FI
157370            FI
157370            IF REMBYT=0 OR A=-1 GO NXMES               % CURRENT MESSAGE EMPTY
157375
157375    % DATA-MESSAGE WITH AT LEAST ONE BYTE
157375
157375            DBCOU GOSW BYT1,BYT2,BYT3,BYT4,BYT5,BYT6,BYT7,BYT8
157407
157407    BYT1:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DA
157414            MIN DBCOU
157415    BYT2:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DA
157421            A\/LAST; *IRW ALEVB DA
157423            MIN DBCOU
157424    BYT3:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DD
157431            MIN DBCOU
157432    BYT4:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DD
157436            A\/LAST; *IRW ALEVB DD
157440            MIN DBCOU
157441    BYT5:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DL
157446            MIN DBCOU
157447    BYT6:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DL
157453            A\/LAST; *IRW ALEVB DL
157455            MIN DBCOU
157456    BYT7:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DX
157463            MIN DBCOU
157464    BYT8:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DX
157470            A\/LAST; *IRW ALEVB DX
157472
157472    % ALL CHARACTERS COLLECTED, RETURN TO USER
157472            0=:DBCOU; A:=10; *IRW ALEVB DT
157475            GO MOURET
157476
157476    % DETERMINE ACTION ON ERRORS
157476    ERR:    IF A<0 GO CNXE                             % XMSG ERROR
157477            IF A=0 GO NXMES                            % MESSAGE EMPTY
157500            IF A=1 OR A=2 THEN
157506               CALL SNDRFI; GO CNXE; GO RETU           % BUFFER EMPTY, SEND RFI
157511            FI
157511            IF A=3 THEN
157514               CALL SNDREJ; GO CNXE; A:=TER01; GO EROUT        % MESSAGE REJECTED
157520            FI
157520            GO RETU
157521
157521    CNXE:   CALL CNVERR
157522    EROUT:  A=:DERROR; 0=:DBCOU
157524    RETU:   GO WDX
157525    RBUS
157541
157541    %=====================================================================
157541    % XX.XX       B B 8 I N P
157541    %
157541    % MON 310: MONITOR CALL RETURNS UP TO 8 BYTES FROM A TAD.
157541    % NUMBER OF BYTES READ IS RETURNED IN T-REG. WHEN LAST
```

```
157541    % CHARACTER IS READ FROM BUFFER T BIT 17 IS SET TO INDICATE
157541    % BREAK CHARACTER.
157541    % CALLED WITH PAGING INTERUPT OFF WITH B = INPUT DATAFIELD
157541
157541    SUBR BB8INP
157541    BB8INP: CALL BRSTACK; CALL IEDCHK; GO EROUT        % INPUT WHILE DELAYED ESCAPE
157544            IF PORTNO=0 THEN A:=TER02; GO EROUT FI     % NOT CONNECTED
157550            0=:DBCOU                                   % ZERO CHARACTER COUNT
157551    POSDA:  IF CURMES><7BDAT THEN                      % NOT DATA MESSAGE
157555    NXMES:     CALL GETMES; GO ERR                     % GET NEW MESSAGE
157557               IF A><X:=7BDAT THEN                     % CONTROLL MESSAGE
157562                  CALL CTRMES; GO CNXE; GO POSDA       % PERFORM CONTROLL
157565               FI
157565            FI
157565            IF REMBYT=0 OR A=-1 GO NXMES               % CURRENT MESSAGE EMPTY
157572
157572    % DATA-MESSAGE WITH AT LEAST ONE BYTE
157572
157572    BYT1:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DA
157577            MIN DBCOU
157600    BYT2:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DA
157604            A\/LAST; *IRW ALEVB DA
157606            MIN DBCOU
157607    BYT3:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DD
157614            MIN DBCOU
157615    BYT4:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DD
157621            A\/LAST; *IRW ALEVB DD
157623            MIN DBCOU
157624    BYT5:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DL
157631            MIN DBCOU
157632    BYT6:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DL
157636            A\/LAST; *IRW ALEVB DL
157640            MIN DBCOU
157641    BYT7:   CALL BYTGET; GO ERR; A=:LAST SHZ 10; *IRW ALEVB DX
157646            MIN DBCOU
157647    BYT8:   CALL BYTGET; GO ERR; A=:LAST; *IRR ALEVB DX
157653            A\/LAST; *IRW ALEVB DX
157655            MIN DBCOU
157656            IF REMBYT=-1 THEN DBCOU BONE 17=:DBCOU FI % LAST CHARACTER, BREAK
157665
157665    % ALL CHARACTERS COLLECTED, RETURN TO USER
157665    OKRET:  A:=DBCOU; 0=:DBCOU; *IRW ALEVB DT
157670            GO MOURET
157671
157671    % DETERMINE ACTION ON ERRORS
157671    ERR:    IF A<0 GO CNXE                             % XMSG ERROR
157672            IF A=0 GO NXMES                            % MESSAGE EMPTY
157673            IF A=1 OR A=2 THEN
157701               IF DBCOU><0 THEN                        % CHARACTERS TO RETURN
157703                  A BONE 17=:DBCOU; GO OKRET           % MARK FOR BREAK
157706               ELSE
157707                  CALL SNDRFI; GO CNXE; GO RETU        % BUFFER EMPTY, SEND RFI
157712               FI
157712            FI
157712            IF A=3 THEN
157715               CALL SNDREJ; GO CNXE; A:=TER01; GO EROUT        % MESSAGE REJECTED
157721            FI
157721            GO RETU
157722
157722    CNXE:   CALL CNVERR
```

```
157723    EROUT: A=:DERROR; O=:DBCOU
157725    RETU:  GO WDX
157726    RBUS
157742
157742
157742    %=============================================================================
157742    %       C N V E R R
157742    %
157742    %       SUBROUTINE TO CONVERT FROM XMSG ERROR
157742    %       TO STANDARD ERROR CODE.
157742    SUBR CNVERR
157742    CNVERR: A-\/XKXXX; EXIT
157745    RBUS
```

```
157746
157746   %=====================================================================
157746   %  B U F F E R   C O N T R O L L   R O U T I N E S   F O R   B A D
157746   %=====================================================================
157746
157746   % XX.XX       P U T P O O L
157746   %
157746   % ROUTINE TO PUT A BUFFER IN FREE POOL OF OUTPUT DATAFIELD
157746   % THE ROUTINE EXECUTES IN IOF TO AVOID DOUBLE ACCESSES TO THE POOL
157746   % ENTRY:     AD-REG - BUFFER-ADDRESS
157746   %             X-REG - BUFFER ID
157746   %             B-REG - TAD OUTPUT DATAFIELD
157746   SUBR PUTPOOL
157746   INTEGER XREG
157747   PUTPOOL: *IOF
157750           IF X=0 GO RETU; X=:XREG                    % DUMMY CALL
157752           A=:T; D=:X; AD:=POOLLI; *STDTX             % PUT OLD POOLLI IN NEW BUFFER
157756           X+2; A:=XREG; *STATX                       % PUT BUFFID IN NEW BUFFER
157761           T=:A; X-2=:D; AD=:POOLLI                   % PUT NEW BUFFER IN POOLLI
157765   RETU:   *ION; EXIT
157767   RBUS
157767
157767   %=====================================================================
157767   % XX.XX       G E T P O O L
157767   %
157767   % ROUTINE TO GET A BUFFER FROM FREE-POOL
157767   % THE ROUTINE EXECUTES IN IOF TO AVOID DOUBLE ACCESSES TO THE POOL
157767   % ENTRY:        B-REG - TAD OUTPUT DATAFIELD
157767   % SKIP RETURN: OK
157767   %               FOLLOWING VARIABLES IN OUTPUT DATAFIELD IS MODIFIED:
157767   %               BUFFID
157767   %               TDTADD
157767   %               TDBTPT
157767   %               REMSIZ
157767   %
157767   % RETURN:       ERROR   A-REG < 0 XMSG ERROR , ELSE POOL EMPTY
157767
157767   SUBR GETPOOL
157767   GETPOOL: *IOF
157770           POOLLI; IF A=0 AND D=0 GO NOPOL            % POOL EMPTY
157774           A=:T; D=:X; *LDDTX                         % GET NEXT IN CHAIN
157777           AD=:POOLLI                                 % UPDATE POOL POINTER
160000           T=:A; X=:D; AD=:TDTADD; X+2; *LDATX        % SAVE ADDRESS AND GET BUFFID
160005           A=:BUFFID; BUDIS=:TDBTPT                   % SET BUFFID AND BYTE POINTER
160010           DFOPP.FBSIZ-BUDIS=:REMSIZ                  % SET REMAINING SIZE
160014           *ION; EXIT AD1
160016   NOPOL:  AD=:TDTADD; 0=:BUFFID=:REMSIZ=:TDBTPT; *ION; EXIT
160024   RBUS
160024
160024   %=====================================================================
160024   % XX.XX       M O V I T O
160024   %
160024   % ROUTINE TO RELOCATE INPUT BUFFER AS OUTPUT BUFFER
160024   % CALLED WITH PAGING OFF
160024   % ENTRY:        B-REG - OUTPUT-DATAFIELD
160024   SUBR MOVITO
160024   MOVITO: DFOPP.BUFFID=:BUFFID; X.TDTADD=:TDTADD
160031           X.FBSIZ-BUDIS=:REMSIZ; BUDIS=:TDBTPT; 0=:CURMES
160037           A:=0=:D; AD=:X.TDTADD
160042           0=:X.BUFFID=:X.REMBYT=:X.TDBTPT=:X.REMSIZ=:X.CURMES
```

```
160047            EXIT
160050    RBUS
160050
160050    %=========================================================================
160050    % XX.XX       C L I D A T
160050    %
160050    % ROUTINE TO REPLACE DATA MESSAGES IN INPUT BUFFER WITH DUMMY
160050    % ENTRY          B-REG - INPUT DATAFIELD
160050    % SKIP RETURN: OK       A-REG = 0  BUFFER IS EMPTY
160050    %                       A-REG = 1  CONTROLL MESSAGES LEFT IN BUFFER
160050    % RETURN:       ERROR    A-REG - XMSG ERROR CODE
160050    SUBR CLIDAT
160050    DISP -3; INTEGER CTRBUF,BUFBP,BREMAI; PSID
160050    CLIDAT: L=:D; CALL SETSTACK(CTRBUF); A:=0; CALL STAB(CTRBUF)
160056            IF BUFFID=0 OR REMSIZ=0 GO OKOUT
160063            REMSIZ; CALL STAB(BREMAI); TDBTPT; CALL STAB(BUFBP)
160071            IF CURMES=7BDAT THEN
160075               T:=REMBYT-; T-1; TDBTPT+T=:TDBTPT; CALL STAB(BUFBP)
160105               REMSIZ-T=:REMSIZ; CALL STAB(BREMAI)
160112            FI
160112            DO WHILE REMSIZ>0
160115               CALL GETMES; GO ECHECK; T=:D
160120               IF CURMES=7BDAT THEN
160124                  T:=TDTAFI; X:=TDTALA; TDBTPT-2=:L SHZ -1; X+A; *LDATX
160134                  IF L BIT "0" THEN
160136                     T:=7DUMM; A/\177400\/T
160141                  ELSE
160142                     T:=7DUMM SHZ 10; A/\377\/T
160146                  FI
160146                  T:=TDTAFI; *STATX
160150               ELSE
160151                  A:=1; CALL STAB(CTRBUF)
160154               FI
160154               TDBTPT+D=:TDBTPT
160157            OD
160160    EOUT:   CALL LDAB(BUFBP); A=:TDBTPT
160163            O=:CURMES=:REMBYT; CALL LDAB(BREMAI); A=:REMSIZ
160170    OKOUT:  CALL LDAB(CTRBUF); GO T2BEXI
160173
160173    ECHECK: IF A<0 GO T1BEXI                          % XMSG ERROR CODE
160175            IF A=3 THEN A:=TER01; GO T1BEXI FI        % MESSAGE ERROR
160202            GO EOUT
160203    RBUS
160214
160214    %=========================================================================
160214    % XX.XX       C L O D A T
160214    %
160214    % ROUTINE TO REPLACE DATA MESSAGES IN OUTPUT BUFFER WITH DUMMY
160214    % ENTRY          B-REG - OUTPUT DATAFIELD
160214    % SKIP RETURN: OK       A-REG = 0  BUFFER IS EMPTY
160214    %                       A-REG = 1  CONTROLL MESSAGES IN BUFFER
160214    % RETURN:       ERROR    A-REG - XMSG ERROR CODE
160214    SUBR CLODAT
160214    DISP -3; INTEGER CTRBUF,BUFBP,AUXBP; PSID
160214    CLODAT: L=:D; CALL SETSTACK(CTRBUF); A:=0; CALL STAB(CTRBUF)
160222            IF BUFFID=0 OR REMSIZ=DFOPP.FBSIZ-BUDIS GO OKOUT
160232            TDBTPT=:D; CALL STAB(BUFBP); BUDIS=:TDBTPT
160240    NXTBY:  DO WHILE TDBTPT<D
160243               CALL LOADBYT; IF A=0 GO NXTBY; A=:CURMES; CALL LOADBYT; A=:X
160250               IF CURMES=7BDAT THEN
```

```
160254                  TDBTPT-1=:TDBTPT; A:=7DUMM; CALL STORBYT
160261               ELSE
160262                  1=:CTRBUF
160264               FI
160264               TDBTPT+X=:TDBTPT
160267            OD
160270            CALL LDAB(BUFBP); A=:TDBTPT
160273   OKOUT:   0=:CURMES=:NOBDIS; CALL LDAB(CTRBUF); GO T2BEXI
160300   RETU:    GO T1BEXI
160301   RBUS
160310
160310   %=====================================================================
160310   % XX.XX        S N D B U F
160310   %
160310   % ROUTINE TO SEND OUTPUT BUFFER
160310   % ENTRY:       B-REG - TAD OUTPUT DATAFIELD
160310   SUBR SNDBUF
160310   SNDBUF: L=:D; CALL SETSTACK(0)
160313            IF BUFFID><0 THEN
160315               IF REMSIZ><DFOPP.FBSIZ-4 THEN
160323                  1=:DRFUNC; CALL STDEV              % SEND BUFFER
160326               FI
160326            FI
160326            GO T2BEXI
160327   RBUS
160331
160331
```

```
160331
160331   %=====================================================================
160331   %      M E S S A G E   G E N E R A T I N G   R O U T I N E S
160331   %
160331   % XX.XX       S N D R F I
160331   %
160331   % ROUTINE TO GENERATE AND SEND RFI-MESSAGE.
160331   % CALLED FROM ALL INPUT CALLS WHEN INPUT BUFFER IS EMPTY.
160331
160331   % ENTRY:           B-REG - TAD INPUT DATAFIELD
160331   % SKIP RETURN:     OK
160331   % RETURN:          A-REG - XMSG ERROR CODE
160331   SUBR SNDRFI
160331   SNDRFI: L=:D; CALL SETSTACK(0); B=:X; DFOPP=:B
160337           IF X.ISTATE<0 THEN                          % NOWAIT MODE
160341              IF X.DFLAG BIT 5RQI GO NOSND             % RFI ALREADY SENT
160344           FI
160344           IF BUFFID=0 AND DFOPP.BUFFID><0 GO USINP    % OUTPUT BUFFER PRESENT?
160351           A:=7RFI; T:=0; CALL CREMES; GO NOSPA        % GENERATE MESSAGE
160355           GO SBUFF
160356
160356   NOSPA:  IF A<0 GO RETU                               % XMSG ERROR
160357           IF A=1 GO NOPOL                             % NO BUFFERS IN POOL
160362           CALL SNDBUF; GO RETU                        % OUTPUT BUFFER FULL, SEND
160364           IF BUFFID=0 GO USINP                        % NO BUFFER, USE INPUT
160366           A:=7RFI; T:=0; CALL CREMES; GO RETU         % GENERATE MESSAGE
160372
160372   SBUFF:  CALL SNDBUF; GO RETU                         % SEND BUFFER
160374           DFOPP.DFLAG BONE 5RQI=:X.DFLAG              % MARK RFI SENT IN INPUT-DATAFIELD
160400   NOSND:  IF DFOPP.BUFFID><0 THEN
160403              0=:X.BUFFID=:X.REMSIZ=:X.CURMES=:X.REMBYT
160407              A=:T; AD:=X.TDTADD; T=:X; CALL PUTPOOL % RETURN INPUT BUFFER TO POOL
160413           FI
160413   OKRET:  DFOPP=:B; GO T2BEXI                          % OK RETURN
160416
160416   USINP:  IF DFOPP.BUFFID=0 GO NOPOL                   % POOL IS EMPTY
160421           CALL MOVITO                                 % MOVE INP. BUFF. TO OUTPUT
160422           A:=7RFI; T:=0; CALL CREMES; GO RETU         % GENERATE MESSAGE
160426           CALL SNDBUF; GO RETU                        % SEND MESSAGE
160430           DFOPP.DFLAG BONE 5RQI=:X.DFLAG              % MARK RFI SENT IN INPUT-DATAFIELD
160434           GO OKRET                                    % OK RETURN
160435   RETU:   X:=DFOPP=:B; GO T1BEXI
160440
160440   NOPOL:  DFOPP.DFLAG BZERO 5RQI BONE 5WRQI=:X.DFLAG
160445           GO OKRET                                    % DRIVER SENDS RFI WHEN BUFFER AVAILABLE
160446   RBUS
160455
160455   %=====================================================================
160455   % XX.XX       S N D R E J
160455   %
160455   % ROUTINE TO GENERATE AND SEND REJECT-MESSAGE.
160455   % IF REJECTED MESSAGE IS DATA, RFI-MESSAGE
160455   % IS ALSO SENDT.
160455
160455   % ENTRY:           B-REG - TAD INPUT DATAFIELD
160455   % SKIP RETURN:     OK
160455   % RETURN:          A-REG - XMSG ERROR CODE
160455   SUBR SNDREJ
160455   SNDREJ: L=:D; CALL SETSTACK(0); A:=DFOPP=:B
160462           IF BUFFID=0 AND DFOPP.BUFFID><0 GO USINP     % OUTPUT BUFFER PRESENT?
```

```
160467          IF DFOPP.CURMES=7BDAT THEN T:=3 ELSE T:=1 FI % INCLUDING RFI?
160477          A:=7REJE; CALL CREMES; GO NOSPA              % GENERATE REJECT
160502          DFOPP.CURMES; CALL BYTPUT; GO RETU           %
160506          A:=7RFI; T:=0; CALL CREMES; GO RETU          % GENERATE RFI
160512          GO SBUFF
160513
160513  NOSPA:  IF A<0 GO RETU                               % XMSG ERROR
160514          CALL SNDBUF; GO RETU                         % OUTPUT BUFFER FULL, SEND
160516          IF BUFFID=0 GO USINP                         % NO BUFFER, USE INPUT
160520          A:=7REJE; T:=1; CALL CREMES; GO RETU         % GENERATE REJECT
160524          DFOPP.CURMES; CALL BYTPUT; GO RETU
160530          IF DFOPP.CURMES=7BDAT THEN
160535             A:=7RFI; T:=0; CALL CREMES; GO RETU       % GENERATE RFI
160541          FI
160541
160541  SBUFF:  CALL SNDBUF; GO RETU                         % SEND BUFFER
160543          IF DFOPP.BUFFID><0 THEN
160546             0=:X.BUFFID=:X.REMSIZ=:X.CURMES=:X.REMBYT
160552             A=:T; AD:=X.TDTADD; T=:X; CALL PUTPOOL    % RETURN INPUT BUFFER TO POOL
160556          FI
160556  OKRET:  DFOPP=:B; GO T2BEXI                          % OK RETURN
160561
160561  USINP:  IF DFOPP.BUFFID=0 THEN                       % NO BUFFERS AVAILABLE
160564             IF X.CURMES=7BDAT THEN
160570                X.DFLAG BONE 5WRQI=:X.DFLAG            % RFI MUST BE SENT BY DRIVER
160573             FI
160573             GO OKRET                                  % REJECT IS NOT POSSIBLE
160574          FI
160574          CALL MOVITO                                  % MOVE INP. BUFF. TO OUTPUT
160575          A:=7REJE; T:=1; CALL CREMES; GO RETU         % GENERATE REJECT
160601          DFOPP.CURMES; CALL BYTPUT; GO RETU
160605          IF DFOPP.CURMES=7BDAT THEN
160612             A:=7RFI; T:=0; CALL CREMES; GO RETU       % GENERATE RFI
160616          FI
160616          CALL SNDBUF; GO RETU                         % SEND BUFFER
160620          GO OKRET                                     % OK RETURN
160621
160621  RETU:  X:=DFOPP=:B; GO T1BEXI
160624  RBUS
160635
160635  %=================================================================
160635  %              C B R E C T A
160635  %
160635  % ROUTINE TO COPY BREAK AND ECHO TABLES
160635  % ENTRY:     AD-REG   - SOURCE ADDRESS
160635  %            B-REG    - TAD OUTPUT DATAFIELD
160635  % RETURN:    EXIT     TDBTPT, REMSIZ IS UPDATED
160635  %
160635  SUBR CBRECTA
160635  CBRECTA: D=:X:=L; CALL SETSTACK(0); X=:D; A=:L
160643          TDBTPT SHZ -1+TDTALA
160646          A=:OBCOU
160647          FOR X:=0 TO 7 DO; X=:XRSA
160654             L=:T; D=:X; *LDATX
160657             D+1; T:=TDTAFI; X:=OBCOU; *STATX
160663             MIN OBCOU
160664          X:=XRSA; OD
160667          TDBTPT+20=:TDBTPT; REMSIZ-20=:REMSIZ
160675          GO T1BEXI
160676  RBUS
```

```
160700
160700   %=================================================================
160700   % XX.XX       B D E C H O
160700   %
160700   % ROUTINE TO SEND ECHO-MESSAGE
160700   % ENTRY:        A-REG = ECHO STRATEGY
160700   %               X-REG = TAD INPUT DATAFIELD
160700   SUBR BDECHO
160700   DISP -3; INTEGER AREG,BREG,MSSIZ; PSID
160700   BDECHO: A=:T; B=:A:=X; L=:D; CALL SETSTACK(AREG);
160706            CALL STAB(BREG); CALL STTB(AREG); DFOPP=:B
160714           IF T=X:=7 THEN T:=21 ELSE T:=1 FI; CALL STTB(MSSIZ)
160724           IF BUFFID=0 THEN
160726              CALL GETPOOL; GO NYTRY
160730           ELSE
160731              IF REMSIZ<T+3 THEN
160735                 CALL SNDBUF; GO NYTRY
160737                 IF BUFFID=0 GO NYTRY
160741              FI
160741           FI
160741           A:=7ECKM; CALL LDTB(MSSIZ); CALL CRHEOD; GO RETU     % CREATE MESSAGE HEADER
160746           CALL LDAB(AREG); A=:D; CALL STORBYT                   % WRITE ECHO TYPE
160752           IF D=7 THEN                                          % COPY ECHO TABLE
160755              DFOPP+"PECH7"=:D:=0; CALL CBRECTA
160762           FI
160762   RETU:   DFOPP=:X; CALL LDAB(BREG); A=:B; GO T1XEXI
160770
160770   NYTRY: DFOPP=:B; RTREF=:X; CALL LDAB(BREG); A=:D; GO REDOM
161000   RBUS
161016
161016   %=================================================================
161016   % XX.XX       B D B R E A
161016   %
161016   % ROUTINE TO SEND BREAK-MESSAGE.
161016   % ENTRY:        A-REG = BREAK STRATEGY
161016   %               X-REG = TAD INPUT DATAFIELD
161016   SUBR BDBREA
161016   DISP -4; INTEGER AREG,BREG,MSSIZ,BRSTR; PSID
161016   BDBREA: A=:T; B=:A:=X; L=:D; CALL SETSTACK(AREG)
161024           CALL STAB(BREG); CALL STTB(AREG)
161030           IF T>X:=7 THEN T:=7 FI; CALL STTB(BRSTR)
161036           IF T=X:=7 THEN T:=23 ELSE T:=3 FI; CALL STTB(MSSIZ); DFOPP=:B
161050           IF BUFFID=0 THEN
161052              CALL GETPOOL; GO NYTRY
161054           ELSE
161055              IF REMSIZ<T+3 THEN
161061                 CALL SNDBUF; GO NYTRY
161063                 IF BUFFID=0 GO NYTRY
161065              FI
161065           FI
161065           A:=7BMMX; CALL LDTB(MSSIZ); CALL CRHEOD; GO RETU     % CREATE MESSAGE HEADER
161072           CALL LDAB(BRSTR); A=:D; CALL STORBYT                  % WRITE BREAK TYPE
161076           TDBTPT SHZ-1; T:=TDTALA+A; DFOPP.BRKMAX
161104           T=:X:=TDTAFI; *STATX                                 % WRITE MAXBREAK
161107           TDBTPT+2=:TDBTPT; REMSIZ-2=:REMSIZ
161115           IF D=7 THEN
161120              CALL LDAB(AREG)
161122              IF A=11 THEN DFOPP.BRKTAB ELSE DFOPP+"PBRK7" FI
161132              A=:D:=0; CALL CBRECTA
161135           FI
```

```
161135   RETU:  DFOPP=:X; CALL LDAB(BREG); A=:B; GO T1XEXI
161143
161143   NXTRY: DFOPP=:B; RTREF=:X; CALL LDAB(BREG); A=:D; GO REDOM
161153   RBUS
161171
161171   %=================================================================
161171   % XX.XX       N W S T A
161171   %
161171   % ROUTINE TO SEND NOWAIT STATUS, CALLED WITH PAGING INTERUPT OFF
161171   % ENTRY:       A = NOWAIT STATUS
161171   %              X = TAD INPUT-DATAFIELD
161171   SUBR NWSTA
161171   DISP -2; INTEGER BREG,NWS; PSID
161171   NWSTA: A=:D; IF X.PORTNO=0 THEN EXITA FI; D=:A                % NOT CONNECTED
161176          L=:D; X:=:B; CALL MLVLOC; CALL BRSTACK
161202          CALL SETSTACK(BREG); CALL STXB(BREG); CALL STAB(NWS)
161210   NXBFF: IF A=0 THEN A:=7NOWT ELSE A:=7TNOW FI                  % DETERMINE NOWT TYPE
161214          T:=DFOPP=:B; T:=1; CALL CRHEEV; GO ECHK               % CREATE MESSAGE
161221          CALL LDAB(NWS); CALL STORBYT                          % WRITE STATUS
161224          CALL SNDBUF; 0/\0; GO RETUO                           % SEND MESSAGE
161227
161227   ECHK:  IF A=1 THEN                                           % NO BUFFER
161232             DFOPP=:B; RTREF=:X; CALL LDAB(BREG); A=:D
161241             CALL MLVULOC; GO REDOM                            % TRY LATER
161243          FI
161243          IF A=2 THEN                                          % NOT ENOUGH SPACE
161246             CALL SNDBUF; GO RETUO; DFOPP=:B
161252             CALL LDAB(NWS); GO NXBFF                          % NEXT BUFFER
161255          FI
161255   RETUO: DFOPP=:B
161257   RETUI: B=:X; CALL LDAB(BREG); A=:B; CALL MLVULOC; GO T2XEXI
161265   RBUS
161302
161302   %=================================================================
161302   % XX.XX       S N D C P
161302   %
161302   % ROUTINE TO SEND COMPLETION CODE CALLED FROM BSCPC
161302   % ENTRY:      AD= COMPLETION CODE
161302   %             B = TAD INPUT DATAFIELD
161302   % RETURN:     EXIT, NO ERRORS RAPPORTED
161302
161302   SUBR SNDCP
161302   DISP -3; INTEGER BREG,CPC1,CPC2; PSID
161302   SNDCP: D=:T:=L; CALL SETSTACK(BREG)
161306          CALL STXB(BREG); CALL STAB(CPC1); CALL STTB(CPC2)
161314          DFOPP=:B                          % B-REG - OUTPUT-DATAFIELD
161316   NXBUF: 7CPCO; T:=5; CALL CRHEEV; GO ECHEC % CREATE MESSAGE
161322          TDBTPT SHZ -1; T:=TDTAFI; X:=TDTALA+A
161327          CALL LDAB(CPC1); *STATX
161332          X+1; CALL LDAB(CPC2); *STATX
161336          TDBTPT+4=:TDBTPT; REMSIZ-4=:REMSIZ
161344          CALL SNDBUF; 0/\0
161346   RETU:  DFOPP=:B; CALL LDXB(BREG); GO T1BEXI
161353   ECHEC: IF A=2 THEN CALL SNDBUF; 0/\0; GO NXBUF FI; GO RETU
161362   RBUS
161374
161374   %=================================================================
161374   % XX.XX       C T R M E S
161374   %
161374   % ROUTINE TO TAKE ACTION ON CONTROLL-MESSAGES
```

```
161374   % ENTRY:        B = INPUT DATAFIELD
161374   % SKIP-RETURN: OK
161374   % RETURN:       ERROR A = XMSG ERROR CODE
161374   SUBR CTRMES
161374   CTRMES: L=:D; CALL SETSTACK(0)
161377           IF CURMES=0 THEN
161401   NXMES:     CALL GETMES; GO ECHECK
161403           FI
161403           IF CURMES=7BDAT GO OKRET                    % DATA MESSAGE
161407           IF A=7TMOD THEN                             % TMODE-MESSAGE
161412              CALL BDTMOD; GO RETU; GO NXMES
161415           FI
161415           IF A=7TTYP THEN                             % TTYPE-MESSAGE
161420              CALL BDTTYP; GO RETU; GO NXMES
161423           FI
161423           IF A=7DESC THEN                             % DEF-ESC MESSAGE
161426              CALL BDDESC; GO RETU; GO NXMES
161431           FI
161431           IF A=7DUMM THEN                             % DUMMY MESSAGE
161434              CALL BDDUMM; GO RETU; GO NXMES
161437           FI
161437   SRJE:   CALL SNDREJ; GO RETU; A:=TER01; GO RETU     % MESSAGE REJECTED
161443
161443   ECHECK: IF A=1 OR A=2 GO OKRET                      % EMPTY
161451           IF A=3 GO SRJE                              % SEND REJECT
161454           GO RETU                                     % XMSG ERROR
161455   OKRET:  GO T2BEXI
161456   RETU:   GO T1BEXI
161457   RBUS
161471
161471   %=====================================================================
161471   % XX.XX       B D T M O D
161471   %
161471   % ROUTINE TO SET TERMINAL MODE ON A TAD
161471   % ENTRY:        B = INPUT-DATAFIELD
161471   % SKIP-RETURN: OK
161471
161471   SUBR BDTMOD
161471   BDTMOD: TDBTPT=:D SHZ -1; T:=TDTAFI; X:=TDTALA+A; *LDATX
161500           IF D BIT "0" THEN A/\377 ELSE A SHZ -10 FI; A=:D
161506           REMSIZ-1=:REMSIZ; MIN TDBTPT; 0=:REMBYT
161513
161513   % CAPITAL LETTERS?
161513           T:=DFLAG BZERO 5CAPITAL
161515           IF D BIT "0" THEN T BONE 5CAPITAL FI; T=:DFLAG
161521
161521   % CR DELAY?
161521           T:=TYPRING BZERO 5CRDLY=:TYPRING
161524           IF D BIT 1 THEN
161526              T BONE 5CRDLY=:TYPRING
161530              T:=DFOPP.TYPRING BONE 5CRDLY=:X.TYPRING
161534           FI
161534
161534   % STOP ON FULL PAGE?
161534           0=:DFOPP.SCREEN
161536           IF D BIT 2 THEN MIN X.SCREEN FI
161541
161541   % LOGGOUT ON MISSING CARRIER
161541           T:=FLAGB BZERO 5LBLOG
161543           IF D BIT 3 THEN T BONE 5LBLOG FI
```

```
161546          T=:FLAGB
161547
161547    RETU:   EXITA
161550    RBUS
161551
161551    %=========================================================================
161551    % XX.XX       B D T T Y P
161551    %
161551    % ROUTINE TO SET TERMINAL TYPE ON A TAD
161551    % ENTRY:           B = INPUT-DATAFIELD
161551    % SKIP-RETURN:     OK
161551    SUBR BDTTYP
161551    BDTTYP: TDBTPT=:D SHZ -1; T:=TDTAFI; X:=TDTALA+A; *LDATX
161560          IF D BIT "0" THEN
161562             A SHZ 10=:D; X+1; *LDATX
161566             A SHZ -10+D
161570          FI
161570          A=:CTTYP; TDBTPT+2=:TDBTPT; REMSIZ-2=:REMSIZ; 0=:REMBYT
161600    RETU:   EXITA
161601    RBUS
161601
161601    %=========================================================================
161601    % XX.XX       B D D E S C
161601    %
161601    % ROUTINE TO SET ESCAPE CHARACTER ON A TAD
161601    % ENTRY:          B = INPUT-DATAFIELD
161601    % SKIP-RETURN:    OK
161601    SUBR BDDESC
161601    BDDESC: TDBTPT=:D SHZ -1; T:=TDTAFI; X:=TDTALA+A; *LDATX
161610          IF D BIT "0" THEN A/\377 ELSE A SHZ -10 FI; A=:T
161616          CESCP/\177400+T=:CESCP
161622          MIN TDBTPT; REMSIZ-1=:REMSIZ; 0=:REMBYT
161627    RETU:   EXITA
161630    RBUS
161632
161632    %=========================================================================
161632    % XX.XX       B D D U M M
161632    %
161632    % ROUTINE TO BYPASS DUMMY MESSAGES
161632    % ENTRY:          B = INPUT-DATAFIELD
161632    % SKIP-RETURN:    OK
161632    SUBR BDDUMM
161632    BDDUMM: T:=REMBYT-; T-1; TDBTPT+T=:TDBTPT; REMSIZ-T=:REMSIZ
161643            0=:REMBYT
161644            EXITA
161645    RBUS
```

```
161645
161645   %=====================================================================
161645   %       T A D   D R I V E R
161645   %=====================================================================
161645
161645   % THIS DRIVER HAS EIGTH EXTERNAL ENTRY POINTS.
161645   %       INIBDR:      INITIAL START, CALLED FROM TADADM IN CONNECTION-
161645   %                    PHASE
161645   %       INISND:      INITIAL SEND, CALLED FROM TADADM TO SEND START-
161645   %                    BUFFER
161645   %       BDRINP:      INPUT-DRIVER, ACTIVATED FIRST TIME FROM INIBDR
161645   %                    ELSE ACTIVATED BY XMSG WHEN RECEIVING A BUFFER.
161645   %       BERESP:      ROUTINE TO SEND ESCAPE-RESPONSE, CALLED FROM
161645   %                    ESCAPE HANDLING.
161645   %       STOTAD:      ROUTINE TO STOP A TAD, THE PORT IS DISCONNECTED
161645   %                    IF OPEN.
161645   %       BDDSCN:      ROUTINE TO DISCONNECT A TAD WITHOUT LOGGING OUT.
161645   %       BDROUT:      OUTPUT-DRIVER, ACTIVATED FROM OUTPUT CALLS.
161645   %=====================================================================
161645   %       M E S S A G E   H E A D E R S
161645
161645   INTEGER HIGHT                  % SAVE TYPE OF HIGH PRIORITY MESSAGE
161646   INTEGER RDATR                  % DATA IN RESPONSE MESSAGE
161647   INTEGER EBUFF:=(7DUMM\0)       % DUMMY-MESSAGE 0 (EMPTY BUFFER)
161650   INTEGER BDESC:=(7ESCA\0)       % ESCAPE-MESSAGE
161651   INTEGER RLOCA:=(7RLOC\0)       % REMOTE LOCAL (RUBOUT NORD-NET)
161652   INTEGER BDDIS:=(7DCON\0)       % DISCONNECT-MESSAGE
161653   INTEGER CESCR:=(7CERS\0)       % CESC-RESP MESSAGE
161654   INTEGER RESCF:=(7RECO\0)       % RESET-CONF MESSAGE
161655   INTEGER NWREM:=(7NWRE\0)       % NOWAIT RESTART MESSAGE
161656   INTEGER ISZRS:=(7ISRS\2)       % ISIZE RESPONSE MESSAGE
161657   INTEGER ERRSP:=(7ERRS\2)       % ERROR-RESPONSE MESSAGE
161660   INTEGER TREPS:=(7TREP\2)       % TREPP STATUS MESSAGE
161661
161661   TRIPLE  DMMES:=(7DUMM\0,0,2) % DUMMY MESSAGE
161664   TRIPLE  RFIMS:=(7RFI\0,0,2)  % RFI MESSAGE
161667   TRIPLE  ERESP:=(7ESRS\0,0,2) % ESCAPE RESPONSE BUFFER
161672   TRIPLE  EDRSP:=(7EDRS\0,0,2) % ESCAPE RESPONSE ESCAPE DISABLED BUFFER
161675
161675   @ICR
161675   SUBR    INIBDR,INISND,
161675           BDRINP,BDROUT,
161675           BERESP,STOTAD,BDLOUT,BDDSCN,BDRWT,
161675           FAERR,FAERO;
161675   @CR;
161675
161675   %=====================================================================
161675   %       I N I B D R
161675   %
161675   % INITIAL ENTRY, LEVEL 10:
161675   %
161675   %       PROGRAMMED FROM TADADM WITH B = TAD INPUT-DATAFIELD
161675   %       TO EXECUTE FOLLOWING TASKS:
161675   %               - OPEN A XMSG PORT.
161675   %               - RESERVE BUFFERS ACCORDING TO NOBUFF AND FBSIZ IN
161675   %                 DATAFIELD, ARRANGE FREE-POOL.
161675   %               - UPDATE PORTNO IN DATAFIELD
161675   %               - RETURN TO TADADM
161675
161675   INIBDR: IF PORTNO=0 THEN                                % NO PORT OPENED
```

```
161677              0=:DFOPP.BXTADD; T:=XFOPN; CALL MXMSG; GO IERR    % OPEN PORT
161704              A=:PORTNO; A:=L=:DFOPP.BXTADD                     % UPDATE DATAFIELDS
161710              IF XFOPN=T THEN T:=XENOT; GO IERR; FI             % SPECIAL TEST FOR XTBLOC
161715              A:=0=:D; AD=:X.POOLLI                             % POOL EMPTY
161720              FOR X:=1 TO NOBUFF DO; X=:XRSA
161725                 T:=XFGET; A:=FBSIZ; CALL MXMSG; GO IERR        % RESERVE BUFFERS
161731                 T:=DFOPP=:B; A=:X=:BUFFID                      % B = OUTPUT-DATAFIELD
161735                 CALL DRXACC(XDINF); GO OERR; *ION
161741                 X:=BUFFID; CALL PUTPOOL; 0=:BUFFID; DFOPP=:B   % PUT BUFFER IN FREEPOOL
161746              X:=XRSA; OD
161751           FI
161751           GO BDRWT; *)FILL                                    % LEAVE LEVEL
161756
161756     INISND: DFOPP=:B; CALL GETPOOL; GO OERR; BUFFID
161763           T:=DFOPP=:B; T:=PORTNO=:D
161767           T:=XFSCM; CALL MXMSG; GO IERR                       % SET CURRENT
161772           TAD:=DMMES; T=:X:=XFWHD; CALL MXMSG; GO IERR
161777           AD:=DFOPP.PARTNER; X:=PORTNO
162002           T:=XFSND; CALL MXMSG; GO IERR                       % SEND BUFFER
162005           T:=XFWDF; A:="BDRINP"; CALL MXMSG; GO IERR          % SET "WAKE UP" MODE
162011           FLAGB BZERO 5LSTA=:FLAGB                            % SET LINE OK BIT
162014           CALL BRSTACK                                        % INIT STACKS.
162015           0=:DFOPP.BUFFID; GO BDRINP                          % ENTER RECEIVE MODE
162020
162020     % ----- INITIALIZING ERRORS -----
162020     OERR:  *ION
162020           A=:T; DFOPP=:B
162021     IERR:  T=:DFOPP.DERROR                                     % SAVE ERROR
162024           GO FAERR
162026     *)FILL
162027
162035     %=========================================================================
162035     %       B D R I N P
162035     %
162035     % TAD INPUT DRIVER.  B = TAD INPUT DATAFIELD
162035     %
162035     % BDRINP IS INITIALLY STARTED BY INIBDR
162035     % THE FUNCTION OF THIS DRIVER IS TO RECEIVE BUFFERS FROM XMSG AND TAKE
162035     % PROPER ACTION DEPENDING ON THE CONTENTS IN THE BUFFER
162035     % ACTTIONS MAY BE THE FOLLOWING:
162035     %       - EMPTY BUFFER, RETURN TO POOL
162035     %       - DATA-MESSAGE, RESPONSE-MESSAGE, WAKE USER
162035     %       - CONTROLL-MESSAGE, PROCEED ACCORDINGLY
162035     %
162035     % THE DRIVER IS RESTARTED BY XMSG WHEN A BUFFER IS QUEUED ON THE PORT.
162035     % THE STRATEGY FOR HANDLING MESSAGES DEPENDS ON WETHER THE MESSAGE IS
162035     % A HIGH PRIORITY MESSAGE OR NORMAL PRIORITY MESSAGE
162035     %       - NORMAL:  MESSAGE IS REMOVED FROM PORT QUEUE ONLY IF
162035     %                  CURRENT INPUT-BUFFER IS EMPTIED (BUFFID=0)
162035     %       - HIGH:    MESSAGE IS IMIDEATELY SERVED.
162035     %=========================================================================
162035
162035     BDRINP: *ION
162036              IF PORTNO=0 GO BDRWT; T:=XFPST BONE XFWAK
162043              CALL MXMSG; GO FAERR                               % READ PORT STATUS
162045              IF T=0 GO BDRWT                                    % NO BUFFER
162047              CALL CHPART; GO BDRINP                             % CHECK IF LEGAL PARTNER
162051              IF XMTHI><T GO FAR NORMP                           % NORMAL PRIORITY
162054              T:=XFRCV; A:=PORTNO; CALL MXMSG; GO FAERR          % RECEIVE
162060              IF T=0 GO BDRWT                                    % NO BUFFER
```

```
162062          X:=D=:TMPBUF
162064          T:=XFRHD; A:=TMPBUF; CALL MXMSG; GO FAERR          % READ SIX BYTES
162070
162070  % CHECK IF RESPONSE-MESSAGE OR EMPTY BUFFER
162070          X=:HIGHT                                          % SAVE MESSAGE TYPE
162071          IF X=EBUFF OR X=CESCR OR X=RESCF OR X=ISZRS OR X=ERRSP THEN
162110            IF X=EBUFF THEN DFLAG BZERO 5RQI=:DFLAG FI      % ALLOW RFI IN NOWAIT
162116            IF X=ISZRS OR X=ERRSP THEN                      % MESSAGE WITH DATA
162124               X:=TMPBUF; T:=6; CALL DRXACC(XDGER); GO FAERR; *ION
162132               A SHZ 10=:RDATR
162134               X:=TMPBUF; T:=7; CALL DRXACC(XDGER); GO FAERR; *ION
162142               T:=RDATR+A=:RDATR                            % DATA IN RDATR
162145            FI
162145            IF DFLAG BIT 5WRQI THEN                       % RFI WAITING TO BE SENDT
162150               A BZERO 5WRQI=:DFLAG; GO FAR RETRFI
162153            FI
162153            IF DFOPP.BUFFID=0 THEN                        % PUT BUFFER TO OUTPUT
162156               X:=TMPBUF; CALL CHSIZE; GO FAR CHRESO
162161               X:=TMPBUF; A:=-1; CALL DRXACC(XDSBP); GO FAERR; *ION
162167               X:=TMPBUF; A:=BUDIS; CALL DRXACC(XDSBP); GO FAERR; *ION
162175               AD=:DFOPP.TDTADD; TMPBUF=:X.BUFFID
162201               BUDIS=:X.TDBTPT; FBSIZ-BUDIS=:X.REMSIZ; 0=:TMPBUF
162207               GO FAR CHRESO; *)FILL
162231            ELSE                                          % PUT BUFFER IN POOL
162232               X:=TMPBUF; CALL CHSIZE; GO FAR CHRESO
162235               X:=TMPBUF; T:=DFOPP=:B; CALL PUTPOOL
162241               DFOPP=:B; 0=:TMPBUF; GO FAR CHRESO
162245            FI
162245          FI
162245          IF X=NWREM THEN                                 % NOWAIT RESTART
162250            T:=XFSND; AD:=DFOPP.PARTNER; X:=PORTNO
162254            CALL MXMSG; GO FAERR                          % RETURN MESSAGE
162256            GO FAR DATRES                                 % RESTART USER
162257          FI
162257          IF X=TREPS THEN                                 % TREP STATUS
162262            X:=TMPBUF; T:=6; CALL DRXACC(XDGER); GO FAERR; *ION
162270            A SHZ 10=:RDATR
162272            X:=TMPBUF; T:=7; CALL DRXACC(XDGER); GO FAERR; *ION
162300            T:=RDATR+A=:RDATR                             % DATA IN RDATR
162303            T:=XFSND; AD:=DFOPP.PARTNER; X:=PORTNO
162307            CALL MXMSG; GO FAERR
162311            T:=RDATR; A:=TINFO
162313            IF T BIT 2 THEN A BONE 5BFUL FI               % BUFFER OVERRUN
162316            IF T BIT 3 THEN A BONE 5PAER FI               % PARITY ERROR
162321            IF T BIT 4 THEN A BONE 5FRER FI               % FRAMING ERROR
162324            A=:TINFO
162325            GO FAR BDRINP
162326          FI
162326          GO FAR ESCDIS; *)FILL                           % ESCAPE OR DISCONNECT
162343
162343  % ----- NORMAL PRIORITY -----
162343  NORMP:  IF BUFFID><0 GO BDRWT                           % INPUT-BUFFER NOT EMPTY
162346          DFLAG BZERO 5RQI=:DFLAG                         % ALLOW RFI IN NOWAIT
162351          T:=XFRCV; A:=PORTNO; CALL MXMSG; GO FAERR       % RECEIVE FUNCTION
162355          IF T=0 GO BDRWT
162357          X-BUDIS=:REMSIZ:=D=:BUFFID=:TMPBUF
162364          X:=BUFFID; CALL CHSIZE; GO WRSIZE; AD=:TDTADD; BUDIS=:TDBTPT
162372
162372  % ----- BUFFER RECEIVED -----
162372  NXMES:  CALL GETMES; GO ECHECK
```

```
162374          IF CURMES=7BDAT GO DATRES                       % DATA MESSAGE
162400          IF A=7TMOD THEN                                 % TMODE-MESSAGE
162403             CALL BDTMOD; GO FAERR; GO NXMES
162406          FI
162406          IF A=7TTYP THEN                                 % TTYPE-MESSAGE
162411             CALL BDTTYP; GO FAERR; GO NXMES
162414          FI
162414          IF A=7DESC THEN                                 % DEF-ESC MESSAGE
162417             CALL BDDESC; GO FAERR; GO NXMES
162422          FI
162422          IF A=7DUMM THEN                                 % DUMMY MESSAGE
162425             CALL BDDUMM; GO FAERR; GO NXMES
162430          FI
162430  SRJE:   CALL REJECT; 0/\0; GO BDRINP                   % MESSAGE REJECTED
162433  ECHECK:  IF A=1 OR A=2 GO BEMTY                         % EMPTY
162441           IF A=3 GO SRJE                                 % SEND REJECT
162444           GO FAERR                                       % XMSG ERROR
162445  *)FILL
162460
162460  WRSIZE: 0=:BUFFID=:CURMES=:REMSIZ; GO BDRINP
162464
162464  BEMTY:  X:=BUFFID; 0=:BUFFID=:CURMES=:REMSIZ
162470          CALL CHSIZE; GO BDRINP
162472          T:=DFOPP=:B; CALL PUTPOOL; DFOPP=:B
162477          GO BDRINP
162500
162500  DATRES:  A:=DFOPP.RSPNUM BZERO 17
162503           IF A><7CERS AND A><7RECO AND A><7ISRS THEN      % NOT AWAITING RESPONSE
162514             CALL STBACK; "IORESTART"=:MFUNC; CALL RTACT   % RESTART USER
162520           FI
162520           GO BDRINP; *)FILL
162527
162527  RSPRST:  0=:DFOPP.RSPNUM                                % RESET RESPONSE WAIT
162531           CALL STBACK; "RSRESTART"=:MFUNC; CALL RTACT    % RESTART AFTER RESPONSE
162535           GO BDRINP
162536
162536  CHRESO:  A:=HIGHT; 0=:HIGHT
162540           IF A=CESCR THEN                                % CESC-RESP
162543             IF DFOPP.RSPNUM=7CERS GO RSPRST
162550             GO BDRINP
162551           FI
162551           IF A=RESCF THEN                                % RESET-CONF
162554             IF DFOPP.RSPNUM=7RECO GO RSPRST
162561             GO BDRINP
162562           FI
162562           IF A=ISZRS THEN
162565              DFOPP.RSPNUM=:T; A BZERO 17
162571              IF A=X:=7ISRS THEN                          % WAITING FOR ISIZE-RESPONSE
162574                RDATR=:RTRES.RTDLGADDR.DAREG              % RETURN PARAMETERS
162600                IF T NBIT 17 THEN A=:X.DXREG FI           % IBRSIZ ONLY
162603                GO RSPRST
162604              FI
162604           FI
162604           IF A=ERRSP THEN
162607              IF DFOPP.RSPNUM=7ERRS THEN
162614                IF RTRES.STATUS BIT 5WAIT THEN
162620                   RDATR=:X.RTDLGADDR.DAREG
162623                   IF A=0 THEN X.DPREG+1=:X.DPREG FI
162627                   GO RSPRST
162630                FI
```

```
162630              FI
162630           FI
162630           DFOPP.RSPNUM BZERO 17
162633           IF A><7CERS AND A><7RECO AND A><7ISRS THEN        % NOT AWAITING RESPONSE
162644             IF DFOPP.ISTATE>=0 THEN X=:B
162650                "IORESTART"=:MFUNC; CALL RTACT; DFOPP=:B      % RESTART OUTPUT
162655             FI
162655           FI
162655           GO BDRINP
162656   *)FILL
162672
162672   % ----- RFI MESSAGE SENT FROM DRIVER WHEN BUFFER ARRIVES AND 5WRQI IS SET.
162672   %       BUFFER WAS NOT AVAILABLE FOR NORMAL SENDING.
162672
162672   RETRFI: TAD:=RFIMS; T=:X:=XFWHD; CALL MXMSG; GO FAERR
162677           AD:=DFOPP.PARTNER; X:=PORTNO
162702           T:=XFSND; CALL MXMSG; GO FAERR; GO FAR CHRESO
162706
162706   % ----- ESCAPE OR DISCONNECT, X = MESSAGE HEADER -----
162706
162706
162706   ESCDIS: IF X=BDESC OR X=RLOCA THEN                    % ESCAPE OR REMOTE-LOCAL MESSAGE
162714             IF DFLAG NBIT 5IESC THEN                    % ESCAPE ENABLED
162717                IF X=BDESC THEN
162722                   CESCP/\377=:LAST                      % ESCAPE
162725                ELSE
162726                   IF FLAGB BIT 5LCHAR THEN
162731                      CESCP SHZ-10=:LAST                 % LOCAL CHARACTER
162734                   ELSE
162735                      177=:LAST                          % RUBOUT IN NORD-NET
162737                   FI
162737                FI
162737                DFLAG BZERO 5RQI=:DFLAG                 % ALLOW RFI IN NOWAIT
162742                CALL ESCAPE; 0/\0
162744                TAD:=ERESP; T=:X:=XFWHD
162747                CALL MXMSG; GO FAERR                    % WRITE ESCAPE-RESPONCE
162751                TMPBUF=:ESCBUF; 0=:TMPBUF; GO BDRINP
162755             ELSE                                     % ESCAPE DISABLED
162756                TAD:=EDRSP; T=:X:=XFWHD
162761                CALL MXMSG; GO FAERR                    % WRITE ESCAPE DISABLED RESPONSE
162763                AD:=DFOPP.PARTNER; X:=PORTNO
162766                T:=XFSND; CALL MXMSG; GO FAERR          % SEND RESPONSE
162771                0=:TMPBUF; GO BDRINP
162773             FI
162773           FI
162773           IF X=BDDIS THEN                            % DISCONNECT-MESSAGE
162776              GO STOTAD                               % STOP AND DISCONNECT TAD
162777           FI
162777           A:=X SHZ -10=:CURMES; GO FAR SRJE          % ILLEGAL HIGH-PRIORITY
163003           GO BDRINP
163004   *)FILL
163022
163022   %=================================================================
163022   %       B E R E S P
163022   %
163022   % ESCAPE RESPONSE, ACTIVATED FROM ESCAPE HANDLING WITH B = INPUT DATAFIELD
163022
163022   BERESP: IF ESCBUF><0 THEN
163024              PORTNO=:D; A:=ESCBUF; 0=:ESCBUF
163030              T:=XFSCM; CALL MXMSG; GO FAERR          % SET CURRENT
```

```
163033              T:=XFSND; AD:=DFOPP.PARTNER; X:=PORTNO
163037              CALL MXMSG; GO FAERR                        % SEND RESPONCE
163041            FI
163041            GO BDRINP
163042
163042    %-----------------------------------------------------------------------
163042    %       S T O B A D
163042    %
163042    % ROUTINE ACTIVATED TO STOP A TAD, B = INPUT DATAFIELD
163042    % IF BACKROUND: TLREP ENABLED:  5LSTA AND 5LOGOUT IS SET
163042    %                TLREP DISABLED: TAD IS LOGGED OUT
163042    % TAD RESERVED: 5LSTA IS SET
163042    % THE TADS PORT IS CLOSED IF OPEN
163042
163042    BDTAD: IF DBPROG><0 AND A=RTRES THEN                  % BACKGROUND
163047             IF FLAGB BIT 5TLREP THEN
163052               A BONE 5LOGOUT=:FLAGB                      % PROGRAM IS RESPONSIBLE, LOGOUT AT MON 0
163054               "IORESTART"=:MFUNC; CALL RTACT             % RESTART PROGRAM
163057             ELSE
163060               CALL BDLOUT                                % LOGGOUT USER
163061             FI
163061           FI
163061           IF FLAGB NBIT 5LSTA THEN GO BDDSCN FI        % PORT SHOULD BE CLOSED
163065           GO BDRWT                                     % LEAVE DRIVER LEVEL
163066
163066    %-----------------------------------------------------------------------
163066    %       B D L O U T
163066    %
163066    % SUBROUTINE TO LOG OUT A TAD
163066    % B = INPUT-DATAFIELD
163066
163066    BDLOUT: A:=L=:"DRIVER"
163070          IF DBPROG><0 AND A=RTRES THEN                 % BACKGROUND AND LOGGED ON
163075             DFLAG BZERO 5IESC=:DFLAG                   % ENABLE ESCAPE
163100             0=:ISTATE=:DFOPP.ISTATE                    % RESET ISTATE
163103             DBPROG.STATUS BZERO 5WAIT=:X.STATUS        % CLEAR WAIT BIT
163107             FLAGB/\ESCMASK BONE 5ESCON=:FLAGB          % RESET USER ESCAPE/LOCAL
163113             -1=:LAST; CALL ESCAPE; 0/\0                % LOGOUT USER
163117          FI
163117          GO DRIVER
163120    *)FILL
163132
163132    %-----------------------------------------------------------------------
163132    %       B D D S C N
163132    %
163132    % ROUTINE TO DISCONNECT A TAD IF PORT IS OPEN, B = INPUT-DATAFIELD
163132    % RETURN IS DONE BY LEAVING DRIVER-LEVEL.
163132
163132    BDDSCN: FLAGB BONE 5LSTA=:FLAGB                       % SET LINE DEAD BIT
163135            IF PORTNO><0 AND DFOPP.BXTADD><0 THEN         % PORT IS OPEN
163142              IF TMPBUF><0 THEN
163144                 0=:TMPBUF; T:=XFREL; CALL MXMSG; 0/\0    % RELEASE CURRENT BUFFER
163150              FI
163150              X.BXTADD=:L; 0=:PORTNO=:X.BXTADD           % CLEAR DATAFIELD
163154              0=:BUFFID=:X.BUFFID=:X.POOLP
163157              T:=XFDCT; *MON 2XMSG                       % DISCONNECT
163161            ELSE                                        % PORT IS CLOSED
163162              0=:PORTNO=:DFOPP.BXTADD
163165              0=:X.POOLP=:BUFFID=:X.BUFFID               % CLEAR DATAFIELD
163170            FI
```

```
163170          GO BDRWT
163171
163171    %=====================================================================
163171    %       B D R O U T
163171    %
163171    % TAD OUTPUT DRIVER ACTIVATED FROM STDEV WITH B = TAD OUTPUT-DATAFIELD
163171
163171    BDROUT: IF DRFUNC=0 OR DFOPP.PORTNO=0 GO BDRWT              % DUMMY CALL
163177          IF BUFFID=0 GO OUT                                    % NO BUFFER
163201          0=:DRFUNC; CALL SETSIZE
163203          DFOPP=:B                                              % B = INPUT-DATAFIELD
163205          PORTNO=:D; A:=DFOPP.BUFFID
163211          T:=XFSCM; CALL MXMSG; GO FAERR                        % SET CURRENT
163214          AD:=DFOPP.PARTNER; X:=PORTNO
163217          T:=XFSND; CALL MXMSG; GO FAERR                        % SEND BUFFER
163222          DFOPP=:B; 0=:CURMES
163225          CALL GETPOOL; 0/\0
163227    OUT:  DFOPP=:B; GO BDRINP               % RETURN VIA INPUT-DRIVER
163232
163232    % ----- F A E R R   FATAL ERROR IN DRIVER
163232    FAERO: DFOPP=:B
163234    FAERR: *ION
163235          GO STOTAD
163236
163236    % ----- B D R W T    LEAVE DRIVER LEVEL
163236    BDRWT: *ION
163237          CALL WT10; *JMP *-1
163241
163241    RBUS
163253
163253    %=====================================================================
163253    %       S E T S I Z E
163253    %
163253    % ROUTINE TO SET CORRECT SIZE AND BYTE-POINTER BEFORE SEND
163253    %       ENTRY: B-REG - TAD OUTPUT-DATAFIELD
163253    SUBR SETSIZE
163253    SETSIZE: A:=L=:DFOPP."DRIVER"
163256          A:=DFOPP.FBSIZ-REMSIZ-BUDIS=:REMSIZ                   % UPDATE SIZE
163263          T:=TDTAFI; X:=TDTALA; A:=0; *STATX
163267          X+1; REMSIZ; *STATX
163272          X:=BUFFID; A:=-1; CALL DRXACC(XDSBP); 0/\0; *ION      % SET BYTE POINTER
163300          X:=BUFFID; TDBTPT; CALL DRXACC(XDSBP); 0/\0; *ION
163306          DFOPP."DRIVER"=:P
163311    RBUS
163312
163312    %=====================================================================
163312    %       C H P A R T
163312    %
163312    % ROUTINE TO CHECK IF BUFFER IS SENT FROM PARTNER
163312    % B = INPUT-DATAFIELD, A = SENDERS NUMBER FROM RECEIVE
163312    SUBR CHPART
163312    CHPART: X:=L=:"DRIVER"
163314            IF A=X:=DFOPP.RPORT THEN                            % SENDER OK
163320               MIN "DRIVER"
163321            ELSE                                                % UNAUTORIZED SENDER
163322               DFOPP.BXTADD=:L:=PORTNO; T:=XFRCV; *MON 2XMSG
163330               CALL WT10; IF T<0 GO DRIVER                      % RECEIVE BUFFER
163333               DFOPP.BXTADD=:L:=D; T:=XFREL; *MON 2XMSG
163341               CALL WT10                                        % RELEASE BUFFER
163342            FI
```

```
163342          GO DRIVER
163343    RBUS
163344
163344    %=================================================================
163344    %       C H S I Z E
163344    %
163344    % ROUTINE TO CHECK IF A BUFFER IS OF CORRECT SIZE, IF NOT THE
163344    % BUFFER IS RELEASED.
163344    % ENTRY:          X-REG - BUFFER ID
163344    %                 B-REG - TAD INPUT DATAFIELD
163344    % SKIP RETURN:   BUFFER OK, AD-REG - BUFFER ADDRESS
163344    % EXIT:          WRONG SIZE, BUFFER IS RELEASED
163344    SUBR CHSIZE
163344    INTEGER XREG
163345    CHSIZE: *IOF
163346          A:=L=:"DRIVER"; X=:XREG
163351          CALL DRXACC(XDINF); GO FAERR; X:=XREG
163355          T=:L
163356          IF L=FBSIZ THEN; *ION
163362             MIN "DRIVER"
163363          ELSE
163364             DFOPP.BXTADD=:L; T:=XFREL; XREG; *ION; MON 2XMSG
163373             CALL WT10
163374          FI
163374          GO DRIVER
163375    RBUS
163400
163400    %=================================================================
163400    %       R E J E C T
163400    %
163400    %  ROUTINE TO SEND REJECT MESSAGE. THE ROUTINE HAS THE SAME FUNCTION
163400    %  AS SNDREJ. REJECT IS ONLY CALLED FROM DRIVER.
163400    %       ENTRY:    B-REG  - INPUT DATAFIELD
163400    %                 CURMES - REJECTED MESSAGE
163400    SUBR REJECT
163400    REJECT: A:=L=:"DRIVER"
163402          IF DFOPP.BUFFID=0 THEN
163405             BUFFID; X:=DFOPP=:B
163410             IF A><0 THEN
163411                CALL MOVITO; GO GESND
163413             ELSE
163414                CALL GETPOOL; GO NOPOL; GO GESND
163417             FI
163417          ELSE
163420             IF REMSIZ>=3 GO GESND
163424             CALL SETSIZE; BXTADD=:L; BUFFID; T:=XFSCM; *MON 2XMSG
163432             CALL WT10; IF T>0 GO FAERO; AD:=PARTNER; X:=DFOPP.PORTNO
163440             T:=BXTADD=:L:=XFSND; *MON 2XMSG
163444             CALL WT10; IF T>0 GO FAERO
163447             CALL GETPOOL; GO NOPOL
163451          FI
163451    GESND: A:=7REJE; CALL STORBYT; A:=1; CALL STORBYT
163455          DFOPP.CURMES/\377; CALL STORBYT
163461          CALL SETSIZE; BXTADD=:L; BUFFID; T:=XFSCM; *MON 2XMSG
163467          CALL WT10; IF T>0 GO FAERO; AD:=PARTNER; X:=DFOPP.PORTNO
163475          T:=BXTADD=:L:=XFSND; *MON 2XMSG
163501          CALL WT10; IF T>0 GO FAERO
163504    NOPOL: DFOPP=:B; GO DRIVER
163507    RBUS
163517
```

```
163517   %=====================================================================
163517   %       M X M S G
163517   %
163517   % ROUTINE TO PERFORM XMSG MONITOR CALL, B = INPUT-DATAFIELD
163517   % SKIP RETURN: OK,     RETURN: XMSG ERROR
163517
163517   SUBR MXMSG
163517   MXMSG: A:=:L=:"DRIVER"; A:=L; X=:L
163523          X:=DFOPP.BXTADD; X:=:L; *MON 2XMSG
163527          CALL WT10; IF T>=0 THEN MIN "DRIVER" FI
163533          GO DRIVER
163534   RBUS
163535
163535   *"-BADAD
163535    @EOF
163535
```

```
163535    %
163535    %=============================================================================
163535    %  PIT3-PIT0-CODE
163535    %
163535    %=============================================================================
163535    *"-3PIT
"163535    *)KILL YENDC; YENDC=*
163535    *7ENDC/
075211    *"
"075211    %
075211    *"8UDMA 8PUDM
"075211    *)KILL 3PITU 3POFU
075211    *"8UDMA+8VICO -3PIT
"075211    * 3PITU; *-1/          % MARK UDMA ON PIT3 SEGMENT
075211    *"
"075211
075211    *"8PUDM+8PVIC -3PIT
"075211    *"3PITU+3POFU
"075211    %=============================================================================
075211    %         MONITOR LEVEL ENTRY FOR MON UDMA
075211    %         ALWAYS ON PIT 0
075211    %=============================================================================
075211    SUBR MUDMA,UDMA,PT3US,UDTMO,UWT11
075211
075211    MUDMA: 0=:ALTMASK; *IRR ALEVB DT;
075213           IF A < LUDV OR  A >HUDV THEN "240"; GO MERR; FI
075223           CALL LOGPH; A:=D
075225           IF A=0 THEN "33"; GO MERR; FI
075230           IF A.RTRES >< RTREF THEN "5"; GO MERR; FI
075237           A:=X.DFPNT; CALL SWAPPR;
075241           GO MONEN
075242    MERR:  *IRW ALEVB DA
075243           GO MONEN
075244
075244    % ENTRY FOR LEVEL 1 PIT 0
075244    UDMA:  CALL SPT3PIT                    % SET NORM PIT=3, ALT PIT=0
075245           GO PT3UDMA                      % ON PIT3-SEGMENT
075246
075246
075246    %=============================================================
075246    % ENTRY FOR LEVEL 11 PIT 0
075246    % (STDRIV=USDRV, DRIVER=UDDRV)
075246    %=============================================================
075246    USDRV: T:="UASTR"; GO FELL
075250    UDDRV: T:="UADDR"; GO FELL
075252    UTIMO: T:="UD11T";
075253    FELL:
075253    *"3PITU
"075253           A:=3132    ; *TRR PCR; PON; BSET ZRO     % NORM PIT=3, ALT PIT=0 RING 2
075257    +"3PITU+3POFU
"075257           T=:P                                     % GO TO ROUTINE
075260    UWT11:
075260    +"3PITU
"075260           A:=132     ; *TRR PCR                    % NORM PIT=0, ALT PIT=0 RING 2
075262    +"3PITU+3POFU
"075262           GO WT11
075263
075263    +)FILL
075302    UDTMO:
075302    %=============================================================
```

```
075302    %      TIMEOUT ROUTINE FOR UNIVERSAL DMA INTERFACE LEVEL 1
075302    %=================================================================
075302          * LDA   (UTIMO           % TIME OUT ROUTINE ON LEVEL 11
075303          * IRW    130 DP
075304          * COPY   SB DA
075305          * IRW    130 DB          % SET BREG TO DATAFIELD
075306          * LDA    (4000
075307          * MST    PID             %ENABLE LEVEL 11
075310          * EXIT                   % RETURN
075311    RBUS
075313    *"
"075313
075313
075313    *"8PIOC 8PPIO
"075313    *)KILL 3PITP 3POFP
075313    *"8PIOC 3PIT
"075313    *"8PPIO -3PIT
"075313    *"8PIOC+8PPIO -3PIT
"075313    SUBR PIOCM,APIOCM
075313    %=================================================================
075313    % LEVEL 1 ROUTINE TO CALL PIOC MONITOR CALL, ACTIVATED FROM SWAPREG
075313    % ALWAYS ON PIT 0
075313    %=================================================================
075313    PIOCM:
075313      *"8PIOC -3PIT
"075313      CALL SPT3PIT                  % SET NORM PIT=3, ALT PIT=0
075314      *"8PIOC+8PPIO -3PIT
"075314      * BSET ZRO
075315      GO APIOCM
075316    RBUS
075320    *"
"075320    *"3PITP+3POFP
"075320
075320    *"8PIOC+8PPIO -3PIT
"075320    SUBR PDRIV,APDRIV
075320    %=================================================================
075320    % ROUTINE ON LEVEL 13 TO CALL PIOC DRIVER
075320    % (DRIVER AND STDRIV = PDRIV) ALWAYS ON PIT0
075320    %=================================================================
075320    PDRIV:
075320         *"8PIOC -3PIT
"075320         * PON
075321         *"8PIOC+8PPIO -3PIT
"075321         GO APDRIV
075322    RBUS
075323    *"
"075323    *"8PIOC+8PPIO -3PIT
"075323    SUBR PITIM
075323    %=================================================================
075323    % TIMOUT AFTER REMOVING RESET AND START
075323    % (TMSUB=PITIM) ALWAYS ON PIT0
075323    %=================================================================
075323    PITIM: X := 1002; A:=1; L=:D;
075326           *"8PIOC -3PIT
"075326           CALL SPT3PIT
075327           *"8PIOC+8PPIO -3PIT
"075327           CALL PWRIT;
075330           *"8PIOC -3PIT
"075330           CALL SPT0PIT
075331           *"8PIOC+8PPIO -3PIT
```

```
 075331          D=:P                                    % SET FLAG TO STOP LOOP
075332   RBUS
075336   *'
 075336
075336   *"8PIOC+8PPIO -3PIT
 075336   SUBR PIORE
075336   %=====================================================================
075336   % SPECIAL IO-RESTART ROUTINE FOR PIOC  (MFUNC=PIORE)
075336   % X = DATAFIELD, RTRES = PROGRAM TO BE STARTED
075336   % KPROS = PROCESS TO BE ACTIVATED (RT)
075336   % ALWAYS ON PITO
075336   %=====================================================================
075336   PIORE: X=:B;=KPROS
075340          PWCR BONE BENA=: PWCR; T := HDEV+3; *IOXT          % ENABLE PIOC
075346          CALL RTENTRY
075347          *IOF
075350          "STDRIV"; *IOF; IRW LV12B DT                      % START DRIVER AGAIN
075353          B=:A; *IRW LV12B DB
075355          "SLV12"; *IRW LV12B DP
075357          LV12; *MST PID; PION
075362          CALL STUPR
075363   RBUS
075367   *"
"075367    *"-3PIT
"075367    *)KILL 7ENDC; 7ENDC=*
075367   *YENDC/
163535   *"
"163535    @EOF
163535
```

```
163535  %=============================================================================
163535  %       M R E S - S I N 2 - G E N
163535  %=============================================================================
163535
163535
163535  % LOGICAL NUMBER TABLES
163535
163535  %=============================================================================
163535  % 42.1         D V 0 0 0
163535  %
163535  DV000, 100
163536          0;0                     % 0 - DUMMY
163540
163540          DT01R;DT01W             % 1 - TERMINAL 1
163542  "
163542  DV000+6/DT01W                   % 2 OUTPUT: ERROR DEVICE
163544  "8DLP1+8DVE1
163544  DV000+14/DILP1
163552  "8SM01
163552  DV000+15/IDM01;UDM01            % 6 - SYNCRON MODEM
163554  "8M1U0
163554  DV000+101/MTDI1;MTD01           % 40 - MAGTAPE UNIT 0
163640  "8TR5
163640  DV000+111/DT05R;DT05W           % 44 - TERMINAL 5
163650  "8TR6
163650  DV000+113/DT06R;DT06W           % 45 - TERMINAL 6
163652  "
163652  EN000=*
163652
163652
163652
163652
163652  %=============================================================================
163652  % 42.2         D V 2 0 0
163652  %
163652  DV200, 100
163653  "IND01+IBL01; ID01I;ID010
163655  "IND02+IBL02; ID02I;ID020
163657  "8SIBA
163657  DV200+101/SI01I;SI010
163755          SI02I;SI020
163757          SI03I;SI030
163761          SI04I;SI040
163763          SI05I;SI050
163765          SI06I;SI060
163767  "
163767  EN200=*
163767
163767
163767
163767  %=============================================================================
163767  % 42.3         D V 3 0 0
163767  %
163767  DV300, 100
163770          SEMI1;0
163772          SEMI2;0
163774          SEMI3;0
163776          SEMI4;0
164000          SEMI5;0
```

```
164002
164002  % SEMAPHORES USED BY THE BACKUP-SYSTEM
164002  DV300+167/SEM60;0               % LOGICAL DEVICE NUMBER 373B
164160          SEM61;0
164162          SEM62;0
164164          SEM63;0
164166          SEM64;0
164170  EN300=*
164170
164170
164170
164170  %=============================================================================
164170  % 42.4         D V 4 0 0
164170  %
164170  DV400, 100
164171
164171
164171
164171
164171
164171
164171
164171
164171
164171
164171  % END OF LOG. NO TABLE FOR DIGIO    (KGS)
164171
164171  EN400=*
164171
164171
164171
164171  %=============================================================================
164171  % 42.5         D V 5 0 0
164171  %
164171  DV500, 100
164172          IERRF;OERRF
164174          CLFIE;0
164176  "-8DI1
164176          0;0
164200  "
164200          RTLFI;0
164202          FS4;0
164204          FS5;0
164206          FS6;0
164210          FS7;0
164212
164212  DV500+33/
164224          DF1;0
164226          DF2;0
164230          NAMSE;0
164232          FS20;0                  % GRAFS SEMAPHORE NO. 2
164234          FS21;0
164236  "8MT1
164236  DV500+55/
164246          DF3;0
164250  "-8S3C
164250  DV500+57/FS27;0
164252          ACSEM;0
164254  "8MT1
164254  DV500+141/
```

```
164332          MTFIE;0
164334  "8MT1+8MT2+8MT3+8MT4
164334  DV500+143/
164334          FS61;0                  % (561) ALL MAGTAPES, DIRECTORY LOCK
164336  "
164336  EN500=*
164336
164336
164336
164336  %=============================================================================
164336  % 42.6          D V 6 0 0
164336  %
164336  DV600, 100
164337  "BADAD
164337  DV600+1/BDSEM                   % BADMIN SEMAPHORES
164340  DV600+3/BASEM;0                 %
164343  "
164343  EN600=*
164343
164343
164343  %=============================================================================
164343  % 42.7          D V 7 0 0
164343  %
164343  DV700, 100
164344
164344  EN700=*
164344
164344
164344
164344  %=============================================================================
164344  % 42.8          D 1 0 0 0
164344  %
164344  D1000,100                       % PERIPHERALS
164345  "8F1U0
164345  D1000+1/F1U0I;F1U00
164347  "
164347  E1000=*
164347
164347
164347  %=============================================================================
164347  % 42.9          D 1 1 0 0
164347  %
164347  D1100,100                       % SYSTEM DEVICES
164350  "8BD1
164350  BIGDI;0
164352  "3B1U0+6B1U0+7B1U0+2B1U0+4B1U0+3C1U0+6C1U0+9C1U0+3D1U0+2D1U0+1E1U0+2E1U0+3E1U0+5E1U0+7E1U0+8E1U0+EE1U0+FE1U
164352  F1101;0
164354  F1102;0
164356  "SLP1+SLP12
164356  D1100+75/
164444  F1136;0
164446  F1137;0
164450  "
164450  D1100+105/
164454  F1142;0
164456  F1143;0
164460  F1144;0
164462  % ************
164462  "8FDI1+8BFD1
164462  D1100+113/FDID1;0
```

```
164464          DF9;0
164466    8F100
164466    D1100+121/F1150;0
164472          F1151;0
164474    "8DLP1+8DVE1
164474    D1100+157/DMLP1;0
164530          DF11;0
164532    .
164532    D1100+177/F1177;0
164550    E1100=*
164550
164550    %=============================================================================
164550    % 42.10        D 1 2 0 0
164550    %
164550    D1200,100
164551          F1200;0                 % *** SPECIAL FOR NORD TPS ***
164553          D1201;0
164555    "8LOG
164555    D1200+5/F1202;0               % RT-PROGRAM-LOG
164557    "8HIST
164557    D1200+7/F1203;0               % HISTOGRAM LOCK.
164561    "-8S3C
164561    D1200+11/F1204;0              % LOCK FOR SINTRAN SERVICE PROGRAM
164563    D1200+13/F1205;0              % MAIL
164565    "
164565    D1200+15/DT01R;DT01W          % 1206 - TERMINAL 1
164567    "IBL01
164567    D1200+21/IB01I;IB01O
164573          DF13;0
164575    "
164575    D1200+45/F1222;0              % TERMINATION COMMAND BUFFER LOCK
164617          F1223;0                 % GRAFS SEMAPHORE NO. 1
164621    "8BCH1
164621    D1200+75/BT01R;BT01W          %1236
164647          BT01I;BT01O
164651    "IBL01
164651    D1200+165/DF20;0
164737    "
164737          0;0;0;0;0;0
164745    E1200=*
164745
164745    %=============================================================================
164745    % 42.11        D 1 3 0 0
164745    %
164745    D1300,100
164746    "8LOG
164746    D1300+125/F1352;0
165074    "8C1HD
165074    D1300+141/HDIF1;0             % 1360              HDLC-1 INPUT
165110    D1300+143/0;HDOF1             % 1361              HDLC-1 OUTPUT
165112    "8C2HD
165112    D1300+145/HDIF2;0             % 1362              HDLC-2 INPUT
165114    D1300+147/0;HDOF2             % 1363              HDLC-2 OUTPUT
165116    "8C1X2
165116    D1300+171/X21F1;0             % 1374              X21 LINE 1
165140    "-8C2X2 -8C3X2 -8C4X2
165140          0;0;0;0;0;0             % FREE NUMBERS AT THE END OF 1300 SERIES
165146                                  % (IF NOT USED BY X21 LINE 2-4)
165146    .
165146    E1300=*
```

```
165146
165146
165146
165146  %=============================================================================
165146  % 42.11.01    D 1 4 0 0
165146  %
165146  D1400, 100
165147  "BAD01;D1400+1/BD01R;BD01W              % 1400 - BACKGROUND-ACCESS-DEVICE  1
165151  "
165151  E1400=*
165151
165151
165151
165151  %=============================================================================
165151  % 42.11.02    D 1 5 0 0
165151  %
165151  D1500, 100
165152
165152  9EBAD=*
165152
165152  E1500=*
165152
165152
165152
165152  %=============================================================================
165152  % 42.11.03    D 1 6 0 0
165152  D1600, 100
165153  "8BU0+8BU1+8BU2+8BU3+8BU4+8BU5+8BU6+8BU7
165153  D1600+1/   F1600; 0
165155  "
165155  E1600=*
165155
165155
165155
165155  %=============================================================================
165155  % 42.11.04    D 1 7 0 0
165155
165155  D1700, 100
165156  "8PC01; D1700+1/PI001;PI001         % 1700 PIOC 1
165160  "
165160
165160  "8RFAC
165160  D1700+61/     DFRRT;0                    % 1730 : REMOTE FILE ACCESS, DF DATAFIELD
165240  "8COSP
165240  D1700+63/     0;F1731                    % 1731 : COSMOS SPOOLING PERIPHERAL DEVICE
165242  "8SIBA+8SIBX+8SIBM
165242  D1700+121/DSI0;0                        % 1750 : SIBAS # 0
165300            DSI1;0                        % 1751 : SIBAS # 1
165302            DSI2;0                        % 1752 : SIBAS # 2
165304  "
165304
165304
165304  E1700=*
165304
165304  D2000, 100
165305
165305
165305  E2000=*
165305
165305  D2100, 100
```

```
165306    8UD01+8VI01
165306   D2100+ 1/UDI01;UD001                    % 2100: UNIVERSAL DMA/VICOM INTERFACE 1
165310   "
165310   "8GPIO; D2100+41/ DTGP0;0               % 2120    GPIB INTERFACE  # 0
165350   "
165350   "8COSP; D2100+155/ F2166;0              % 2166    COSMOS SPOOLING    , QUEUE SEMA
165464           D2100+157/ F2167;0              % 2167    COSMOS SPOOLING    , I/O SEMA
165466   "
165466   E2100=*
165466
165466   D2200, 100
165467   "8DILG
165467           DFDIL;0                 % 2200: DISC-ACCESS-LOG DATAFIELD
165471           SEMDL;0                 % 2201: DISC-ACCESS-LOG-BUFFER LOCK
165473   "
165473
165473   E2200=*
165473
165473   D2300, 100
165474   *+NULDN+NULDN/              %  USER  LOGICAL DEVICE NUMBERS
165514   E2300=*
165514
165514   D2400, 100
165515
165515   E2400=*
165515
165515
```

```
165515  %
165515  %=============================================================================
165515  % TIMER TABLE
165515
165515  %=============================================================================
165515  % 42.12       T M R T A
165515  %
165515  TMRTA=*
165515  "8BD1
165515          BIGDI
165516  "
165516  M1TMR, -1
165517  "8DLP1+8DVE1;DMLP1
165520  "8FDI1+8BFD1;FDID1
165521  "8SM01
165521          IDM01;UDM01
165523  "8C1HD;HDOF1
165524  "8C2HD;HDOF2
165525  "8C1X2;X21F1
165526  "8PC01;PI001
165527  "8MT1
165527          MTFIE
165530  "
165530  "8CP51; 5CPU1
165531  "8M100 99SM1; MTDI1
165532  "8PC01; PI001
165533  "8MOLI; MLIDF
165534  "8UD01+8VI01; UDI01;
165535  "8GPI0;DTGP0
165536  "
165536  **XTIME/                        % EXTRA ENTRIES, DEFAULT=6
165544
165544  9SXTD=*                         % START OF TERMINALS
165544          DT01R;DT01W
165546  IMRTE, -1; 0
165550  "8TR5
165550          DT05R;DT05W
165552  "8TR6
165552          DT06R;DT06W
165554  "
165554  9EXTD=*                         % END OF TERMINALS
165554
165554  "8AD01;BD01R;BD01W
165556  "
165556  ETMRT, -1                               % END OF TIMER TABLE
165557
```

```
165557
165557  %=============================================================================
165557  % IDENT CODE TABLES
165557
165557  %=============================================================================
165557  % 42.13       I T B 1 0
165557  %
165557  ITB10, DT01W
165560  "8DLP1+8DVE1;ITB10+2/DMLP1
165562  "8SM01
165562  ITB10+3/UDM01
165563  "8TR5
165563  ITB10+43/DT05W
165623  "8TR6
165623  ITB10+44/DT06W
165624  "
165624  ITB10+117/DT01W                 % 120 - TERM. 1, FOR MULTITERM INTERF.
165677
165677  MAX10=*-ITB10
165677
165677
165677
165677  %=============================================================================
165677  % 42.14       I T B 1 1
165677  %
165677  ITB11=*
165677  "8MT1
165677  ITB11+2/MTFIE
165702  "8BD1
165702  ITB11+16/8IGDI
165716  "8FDI1+8BFD1
165716  ITB11+20/FDID1
165720  "
165720
165720  MAX11=*-ITB11
165720
165720
165720
165720  %=============================================================================
165720  % 42.15       I T B 1 2
165720  %
165720  ITB12, DT01R
165721  "8SM01
165721  ITB12+3/IDM01
165724  "8CP51
165724  ITB12+15/5CPU1
165736  "8TR5
165736  ITB12+43/DT05R
165764  "
165764  "8TR6
165764  ITB12+44/DT06R
165765  "
165765  ITB12+117/DT01R
166040  "8C1HD;ITB12+147/HDOF1
166070  "8C2HD;ITB12+150/HDOF2
166071
166071  MAX12=*-ITB12
166071
166071
166071  %=============================================================================
```

```
166071  % 42.16       I T E 1 0
166071  %
166071  ITE10=*
166071
166071  *+XID10/
166103  -1
166104  %=============================================================================
166104  % 42.17       I T E 1 1
166104  %
166104  ITE11=*
166104  "8UD01; 140010;UDI01;          % UNIVERAL DMA INTERFACE #1
166106  "8GPI0; 140000; DTGP0          % GPIB INTERFACE # 0
166110  "
166110
166110
166110
166110  *+XID11/
166122  -1
166123  %=============================================================================
166123  % 42.18       I T E 1 2
166123  %
166123  ITE12=*
166123  "8PC01; 140002; PI001
166125  "
166125  *+XID12/
166137   1
166140
166140  %=============================================================================
166140  % 42.19       I T E 1 3
166140  %
166140  ITE13=*;
166140  *+XID13/
166144  -1
166145  IT13E=*
166145
166145  %=============================================================================
166145  % 42.20       I T B 1 3
166145  %
166145  ITB13, CLCFI
166146  "8MPM4
166146          ITB13+17/ MPM4D
166165          ITB13+20/ MPM4D
166166          ITB13+21/ MPM4D
166167          ITB13+22/ MPM4D
166170          ITB13+23/ MPM4D
166171          ITB13+24/ MPM4D
166172          ITB13+25/ MPM4D
166173          ITB13+26/ MPM4D
166174          ITB13+27/ MPM4D
166175          ITB13+30/ MPM4D
166176          ITB13+31/ MPM4D
166177          ITB13+32/ MPM4D
166200          ITB13+33/ MPM4D
166201          ITB13+34/ MPM4D
166202          ITB13+35/ MPM4D
166203          ITB13+36/ MPM4D
166204          ITB13+37/ MPM4D
166205          ITB13+40/ MPM4D
166206   "8C1HD;ITB13+147/HDIF1
166315   "8C2HD;ITB13+150/HDIF2
```

```
166316
166316
166316  MAX13=*-ITB13
166316
166316  %=============================================================================
166316  %               I D 1 0 T
166316  %
166316  % IDENT CODE TABLE FOR TERMINAL 65-128 OUTPUT (LEVEL 10)
166316
166316  ID10T=*
166316
166316  MXX10=*-ID10T
166316
166316  %=============================================================================
166316  %               I D 1 2 T
166316  %
166316  % IDENT CODE TABLE FOR TERMINAL 65-128 INPUT (LEVEL 12)
166316
166316  ID12T=*
166316
166316  MXX12=*-ID12T
166316
```

```
166316
166316  %%%%%%%%%%%%%%%%%%%%%%%%%   B U F F E R S   %%%%%%%%%%%%%%%%%%%%%%%%%%%
166316  %=============================================================================
166316  %       LINE-PRINTER AND/OR VERSATEC BUFFERS
166316  %
166316  "8DLP1+8DVE1
166316  LP1BU=*; *+200/
166516  "
166516  "8BACS
166516
166516  %=============================================================================
166516  %               X B C K T
166516  %
166516  % THIS TABLE IS USED TO FIND THE ADDRESS OF THE BACKGROUND PROGRAMS
166516  % INPUT DATAFIELD. (ACTUAL BACKGROUND PROGRAM - BAK01) IS INDEX IN
166516  % THIS TABLE. (USED BY GETDATAFIELD)
166516  %
166516  XBCKT=*
166516  "8BACS; DT01R
166517  "8BP5 8BACS; 0
166520  "8BP6 8BACS; 0
166521  "8BACS 8BCH1;BT01R
166522  "
166522
166522
```

```
166522  )LINE
166522  %=============================================================================
166522  %       M 1 D R I V E R S
166522  %=============================================================================
166522
166522  %=============================================================================
166522  % 43.0         N O R D   1 0 0   D R I V E R S   F O R   D M A   D E V I C E S
166522
166522
166522
166522  )LINE0        N O R D   1 0 0   D R I V E R S   F O R   D M A   D E V I C E S

166522  %=============================================================================
166522  %       M 2 D R I V E R S
166522  %=============================================================================
166522
166522  "8FDI1+8FDI2+8BFD1+8BFD2
166522
166522  )KILL RDAT  RSR1 RSR2 WCWD WDAD WSCT INTEN BBUSY
166522  )KILL ENINT DVCL BCLBU WRDAT WRDEL RDID FREAD SEEK
166522  )KILL RECAL
166522  )KILL 9TREG 9AREG 9DREG 9XREG 9LREG
166522
166522  "
```

```
166522  %
166522
166522  )LINE166522    *"8BFD1+8BFD2
"166522
166522    SUBR BFDIS
166522    %=============================================================================
166522    %  43.9       B F D I S
166522    %
166522    % F L O P P Y   D I S K   D R I V E R   F O R   S I N G L E /
166522    % D O U B L E   D E N S I T Y  &  S I D E
166522    %
166522    % MODIFIED: 3/6 -81
166522    % MODIFIED: 24/2-82 BY TP
166522    %
166522    %        CALLING SEQUENCE:
166522    %        JPL I (BFDIS
166522    %        JMP    ERROR            ERROR EXIT
166522    %        JMP    BUSY             BUSY EXIT
166522    %        JMP    FIN              FINISHED
166522    %
166522    %
166522    %        REGISTER CONTENT WHEN CALLING BFDIS:
166522    %
166522    %        X- NUMBER OF SECTORS TO TRANSFER
166522    %        T- BIT 0-5: FUNCTION CODE
166522    %
166522    %               0 - READ
166522    %               1 - WRITE
166522    %               7 - ERASE TAPE                              *
166522    %              10 - ADVANCE TO EOF
166522    %              11 - REVERCE TO EOF
166522    %              12 - WRITE EOF
166522    %              13 - REWIND TO BOT
166522    %              20 - READ STATUS
166522    %              36 - READ EXTENDED STATUS                    *
166522    %              41 - FORMAT FLOPPY                           *
166522    %              42 - READ FORMAT                             *
166522    %              43 - READ DELETED RECORD
166522    %              44 - WRITE DELETED RECORD
166522    %              54 - COPY FLOPPY                             *
166522    %              55 - FORMAT TRACK (FLOPPY)
166522    %              56 - CHECK CARTRIDGE (READ AND TEST CRC)     *
166522    %              57 - TEST CARTRIDGE CAPASITY (WRITE)         *
166522    %              70 - RETENSION CARTRIDGE                     *
166522    %              71 - PERFORM TEST                            *
166522    %              72 - ILLEGAL COMMAND                         *
166522    %              73 - CLEAR                                   *
166522    %              74 - Continious Read                         *
166522    %              75 - Continious Write                        *
166522    %
166522    %        BIT 6-7: UNIT NUMBER  (0,1,2,3)
166522    %
166522    %        BIT 8-9: BYTES/SECTOR:  0 = 512
166522    %                                1 = 256
166522    %                                2 = 128
166522    %                                3 = 1024
166522    %
166522    %        BIT 10: DOUBLE SIDED DRIVE & DISKETTE
166522    %
166522    %        BIT 11: DUAL DENSITY DISKETTE
```

```
166522   %
166522   %       POSSIBLE FORMATS (BITS 8-9-10-11):
166522   %
166522   %             0- IBM SYS-32-II   512 BYTES/SECTOR    SINGLE SIDE, SINGLE DENSITY
166522   %             1- IBM 3600        256 BYTES/SECTOR    SINGLE SIDE, SINGLE DENSITY
166522   %             2- IBM 3740        128 BYTES/SECTOR    SINGLE SIDE, SINGLE DENSITY
166522   %             3- ILLEGAL
166522   %             4- NON IBM         512 BYTES/SECTOR    DOUBLE SIDE, SINGLE DENSITY
166522   %             5- NON  IBM        256 BYTES/SECTOR    DOUBLE SIDE, SINGLE DENSITY
166522   %             6- NON  IBM        128 BYTES/SECTOR    DOUBLE SIDE, SINGLE DENSITY
166522   %             7- ILLEGAL
166522   %            10- NON  IBM        512 BYTES/SECTOR    SINGLE SIDE, DOUBLE DENSITY
166522   %            11- IBM SYS-34      256 BYTES/SECTOR    SINGLE SIDE, DOUBLE DENSITY
166522   %            12- ILLEGAL
166522   %            13- NON  IBM       1024 BYTES/SECTOR    SINGLE SIDE, DOUBLE DENSITY
166522   %            14- NON  IBM        512 BYTES/SECTOR    DOUBLE SIDE, DOUBLE DENSITY
166522   %            15- IBM SYS-34      256 BYTES/SECTOR    DOUBLE SIDE, DOUBLE DENSITY
166522   %            16- ILLEGAL
166522   %            17- NON IBM        1024 BYTES/SECTOR    DOUBLE SIDE, DOUBLE DENSITY
166522   %
166522   %
166522   %       BIT 12: STREAMER CASSET
166522   %
166522   %       BIT 14-15: DESTINATION UNIT DURING COPY COMMAND
166522   %
166522   %       D- DISK ADDRESS (LOGIC SECTOR ADDRESS)
166522   %
166522   %       ADDRESS RANGE:
166522   %
166522   %             0-1150   FORMAT  0    232      PAGES
166522   %             0-2203   FORMAT  1    220.375  PAGES
166522   %             0-3722   FORMAT  2    175.125  PAGES
166522   %                      FORMAT  3
166522   %             0-2320   FORMAT  4    464      PAGES
166522   %             0-4406   FORMAT  5    440.75   PAGES
166522   %             0-7644   FORMAT  6    372.25   PAGES
166522   %                      FORMAT  7
166522   %             0-2203   FORMAT 10    440.75   PAGES
166522   %             0-3722   FORMAT 11    372.25   PAGES
166522   %                      FORMAT 12
166522   %             0-1150   FORMAT 13    464      PAGES
166522   %             0-4406   FORMAT 14   1101.5    PAGES
166522   %             0-7644   FORMAT 15    764.5    PAGES
166522   %                      FORMAT 16
166522   %             0-2320   FORMAT 17   1150      PAGES
166522   %
166522   %  EXIT INFORMATION:
166522   %
166522   %       IF NORMAL OPERATIONS
166522   %         T-REG = HARDWARE STATUS
166522   %         X-REG = STATUS 1 (FROM FSTAT+0):
166522   %         D-REG = STATUS 2 (FROM FSTAT+1):
166522   %
166522   %       IF TEST OPERATIONS
166522   %         T-REG = HARDWARE STATUS
166522   %         X-REG = HARDWARE STATUS
166522   %         D-REG = 177777
166522   %
166522   %                 STATUS 1 (FROM FSTAT+0):
166522   %                                BIT 0 - NOT USED
```

```
166522   %                              1 - INT. ENABLED
166522   %                              2 - CONTROLLER BUSY
166522   %                              3 - CONTROLLER READY
166522   %                              4 - OR OF ERRORS
166522   %                              5 - DELETED RECORD
166522   %                              6 - INTERNAL RETRIES (NO ERROR)
166522   %                              7 - SERIOUS ERROR, NO MEMORY CONTACT
166522   %                              8 - NOT USED
166522   %
166522   %                        BIT 9-15 - ERROR CODE:
166522   %                              0 - TRANSFER OK
166522   %                              5 - CRC ERROR
166522   %                              6 - SECTOR NOT FOUND
166522   %                              7 - TRACK NOT FOUND
166522   %                             10 - FORMAT NOT FOUND
166522   %                             11 - DISKETTE DEFECT (IMPOSSIBLE TO FORMAT)
166522   %                             12 - FORMAT MISMATCH
166522   %                             13 - ILLEGAL FORMAT SPECIFIED
166522   %                             14 - SINGLE SIDED DISKETTE INSERTED
166522   %                             15 - DOUBLE SIDED DISKETTE INSERTED
166522   %                             16 - WRITE PROTECTED DISKETTE/CARTRIGE
166522   %                             17 - DELETED RECORD
166522   %                             20 - DRIVE NOT READY
166522   %                             21 - CONTROLLER BUSY ON START
166522   %                             22 - LOST DATA (OVER OR UNDERRUN)
166522   %                             23 - TRACK ZERO NOT DETECTED
166522   %                             24 - VCO FREQUENCY OUT OF RANGE
166522   %                             25 - MICROPROGRAM OUT OF RANGE
166522   %                             26 - TIMEOUT
166522   %                             27 - UNDEFINED ERROR
166522   %                             30 - TRACK OUT OF RANGE
166522   %                             32 - COMPARE ERROR (DURING COMPARE OF DATA)
166522   %                             33 - INTERNAL DMA ERRORS
166522   %                             40 - ND-100 BUS ERROR COMMAND FETCH
166522   %                             41 - ND-100 BUS ERROR STATUS TRANSFER
166522   %                             42 - ND-100 BUS ERROR DATA TRANSFER
166522   %                             43 - ILLEGAL COMMAND
166522   %                             44 - WORDCOUNT NOT ZERO
166522   %                             45 - ILLEGAL COMPETION (CONT. TRANSF)
166522   %                             46 - ADR-REG ERROR
166522   %                             50 - NO BOOTSTRAP FOUND ON DISKETTE
166522   %                             51 - WRONG BOOTSTRAP (TOO OLD FLO-MON VERSION)
166522   %                             60 - STREAMER HANDSHAKE ERROR
166522   %                             61 - STREAMER STATUS TRANSFER ERROR
166522   %                             62 - BAD CARTRIGE
166522   %                             63 - NO CARTRIGE INSTALLED
166522   %                             64 - END OF TAPE, CARTRIGE FULL
166522   %                             65 - STREAMER DRIVE ERROR
166522   %                             66 - UNIDENTIFIED EXCEPTION
166522   %                             67 - ILLEGAL COMMAND TO STREAMER
166522   %                             70 - PROM CHECKSUM ERROR
166522   %                             71 - RAM ERROR
166522   %                             72 - CTC ERROR
166522   %                             73 - DMACTRL ERROR
166522   %                             74 - VCO ERROR
166522   %                             75 - FLOPPY CONTROLLER ERROR
166522   %
166522   %
166522   %                  STATUS 2 (FROM FSTAT+1):
166522   %                        BIT   0-3  - IF READ FORMAT COMMAND OR FORMAT ERROR:
```

```
166522   %                    FORMAT READ FROM DISKETTE, OTHERWISE = 0
166522   %                4-5  - NOT USED
166522   %                  6  - TRUE IBM FORMAT (128 B/S ON TR. 0 SIDE 0)
166522   %                  7  - NOT USED
166522   %                8-9  - LAST ACCESSED UNIT
166522   %               10-15 - NOT USED
166522   %
166522   %       HARDWARE STATUS (HDEV+2):
166522   %               BIT 0-6 - EQUAL TO STATUS 1 (FROM FSTAT+0)
166522   %                  7 - SERIOUS ERROR !!!!!!!!!!!!
166522   %                8-14 - ALWAYS 0
166522   %                  15 - ALWAYS 1 ( = NEW CONTROLLER)
```

```
166522
166522    %       USE OF BUFFER WHILE RUNNING TESTS
166522    %
166522    %                 ********************************************
166522    % BUFFER +00      * TEST NO                                   *
166522    % BUFFER +01      * BITS SET COROSPONDING TO IOX'S TO PERFORM *
166522    % BUFFER +02      * DATA READ FROM         IOX DEVNO +0       *
166522    % BUFFER +03      * DATA TO BE WRITHEN TO  IOX DEVNO +1       *
166522    % BUFFER +04      * DATA READ FROM         IOX DEVNO +2       *
166522    % BUFFER +05      * DATA TO BE WRITHEN TO  IOX DEVNO +3       *
166522    % BUFFER +06      * DATA READ FROM         IOX DEVNO +4       *
166522    % BUFFER +07      * DATA TO BE WRITHEN TO  IOX DEVNO +5       *
166522    % BUFFER +06      * DATA READ FROM         IOX DEVNO +4       *
166522    % BUFFER +10      * DATA READ FROM         IOX DEVNO +6       *
166522    % BUFFER +11      * DATA TO BE WRITHEN TO  IOX DEVNO +7       *
166522    %                 ********************************************
166522    %                 *  USE USER SPESIFYED CCB                    *
166522    %                 ********************************************
166522    % BUFFER +12      *  CCBWO,  COMMAND WORD                      *
166522    % BUFFER +13      *  BFDEV,  DEVICE ADDRESS                    *
166522    % BUFFER +14      *  FMEMH,  MEMORY ADDRESS ( HIGH )           *
166522    % BUFFER +15      *  FMEML,  MEMORY ADDRESS ( LOW )            *
166522    % BUFFER +16      *  OPWCH,  OPTIONS AND WORD COUNT ( HIGH )   *
166522    % BUFFER +17      *  WCOUN,  WORD/RECORD COUNT ( LOW )         *
166522    % BUFFER +20      *  FSTA1,  STATUS 1                          *
166522    % BUFFER +21      *  FSTA2,  STATUS 2                          *
166522    % BUFFER +22      *  LASMH,  LAST MEMORY ADDRESS ( HIGH )      *
166522    % BUFFER +23      *  LASML,  LAST MEMORY ADDRESS ( LOW )       *
166522    % BUFFER +24      *  MREMW,  MOUST REMAINING WORDS             *
166522    % BUFFER +25      *  LREMW,  LEAST REMAINING WORDS             *
166522    %                 ********************************************
166522    % BUFFER +26      *  FUTHER EXTENTIONS                         *
166522    % BUFFER +27      *  FUTHER EXTENTIONS                         *
166522    %                 ********************************************
166522    % BUFFER +30      *  REAL BUFFER START                         *
166522    %                 ********************************************
```

```
166522
166522
166522    SYMBOL IOXIN=164000
166522    %SYMBOL RDATA=0                    % READ DATA
166522    SYMBOL RSTAT=2                     % READ STATUS
166522    SYMBOL LCONT=3                     % LOAD CONTROL WORD
166522    SYMBOL LHPOI=5                     % LOAD POINTER HIG
166522    SYMBOL LLPOI=7                     % LOAD POINTER LOW
166522    SYMBOL LDATA=7                     % LOAD DATA
166522
166522    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
166522    %                                  %
166522    %   OPERATION DECLARATIONS         %
166522    %   *):NEW                         %
166522    %                                  %
166522    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
166522
166522    SYMBOL RSSTA=20                    %    READ STATUS
166522    SYMBOL PTEST=71                    % * PERFORM TEST
166522    SYMBOL FCLEAR=73                   % * CLEAR
166522
166522    SYMBOL FOCCU=2, INERR=4, SEERR=7
166522
166522    INTEGER TEARG
166523    INTEGER TETRG
166524
166524    BEDIS:  TAD=:TADRG        ; X=:XRG
166526
166526            IF A:=77 /\ T-FCLEAR >< 0 THEN
166532
166532              A:=HDEV+RSTAT    ; * EXR SA
166535              IF A BIT FOCCU GO HBUSY
166537
166537              X:=A
166540              K:="0"
166541              IF A:=BUSFL><0          THEN K:=1 FI
166544
166544              IF A:=77 /\ T-RSSTA=0 THEN K:=1 FI
166551              IF K THEN
166553
166553                X=:XRG
166554                T:=HDEV ; * EXR ST
166556                D:=A
166557                IF X BIT INERR THEN
166561
166561                  IF A:=TRG NBIT 14  THEN
166564                    T+LCONT
166565                    A:=22 ; * EXR ST
166567                  FI
166567                  L-1
166570
166570                ELSE
166571
166571                  IF A:=77 /\ TRG -PTEST=0 THEN
166575
166575                    X:=HDEV
166576                    AD:=MEMAD ; D+1 ; A:=A+C ; * EXAM
166602                    T BZERO 10
166603                    DO
166603                      D+1 ; A:=A+C
```

```
%
%
%    TEMPRARY SAVED A-REG DORING TEST IOXS
%    TEMPRARY SAVED T-REG DORING TEST IOXS
%
%
%
%
%
%    READ STATUS
%    CONTROLLER BUSY
%
%
%
%    PREVIOUS TRANSFER TO BE CHECKED ?
%    OR
%    OPER. READ STATUS ON FLOPPY?
%
%    YES, PREVIOUS TRANSFER TO BE CHECKED
%
%    READ STATUS 1
%
%    ERRORS
%
%    IF NOT STREAMER
%
%    CLEAR DEVICE
%
%    ERROR RETUR
%
%    NO ERRORS, FINISHED
%
%    IF PERFORME TEST
%
%
%
%
%
%
```

```
166605                    IF T BIT "0" THEN                                  %
166607                      T=:TETRG ; A=:TEARG                              %
166611                      * EXR SX                                         %   EXIQUTE THE INPUT IOX
166612                      T:=A ; A:=TEARG                                  %
166614                      * DEPO                                           %   STORE THE READ DATA
166615                      T:=TETRG                                         %
166616                    FI                                                 %
166616                   WHILE T SHZ -2 ><0                                  %
166621                     X+2 ; D+1 ; A:=A+C                                %
166624                   OD                                                  %
166625                                                                       %
166625                 FI                                                    %
166625                                                                       %
166625   RETFI:        L+1                                                   %   FINISED RETUR
166626                                                                       %
166626               FI                                                      %
166626               T:=FSTA2                                                %   STATUS 2 (ERROR CODE & FORMAT NO.)
166627               X:=XRG:=:D                                              %   HWSTAT IN D, FSTA1 IN X
166631               IF A:=77 /\ TRG -PTEST=0  AND D NBIT INERR THEN         %   IF PREFORME TEST WENT OK
166637                 X:=D                                                  %   HWSTAT TO X
166640                 T:=-1                                                 %
166641               FI                                                      %
166641               T:=:D                                                   %   HWSTAT TO T, FSTA2/-1 TO D
166642               A:=ARG                                                  %
166643               0=:BUSFL                                                %
166644               GO RETUR                                                %
166645                                                                       %
166645   HBUSY:      GO BUSYR                                                %   BUSY RETURN
166646                                                                       %
166646             FI                                                        %
166646           FI                                                          %
```

```
166646
166646
166646          * BLDA 140 DT
166647
166647          IF A:=77 /\ T -PTEST=0 THEN
166653
166653            AD:=MEMAD ; * EXAM
166655            T SHZ 11
166656            T BONE 10 BONE 3
166660            T BONE 1 ; * BSTA 60 DT
166662            T=:SFCOM
166663            X:=HDEV+1
166665            D+1 ; A:=A+C ; * EXAM ; BLDA 100 DT
166671            T BZERO 10
166672            DO
166672              D+1 ; A:=A+C
166674              D+1 ; A:=A+C
166676              IF T  BIT 1 THEN
166700                T=:TETRG
166701                * EXAM ; SWAP SA DT
166703                * EXR SX
166704                T=:A:=TETRG
166706              FI
166706            WHILE T SHZ -2><0
166711              X+2
166712            OD
166713            IF K THEN
166715              T:=HDEV+LLPOI
166717              AD:=MEMAD  ; A:=:D+12          ; * EXR ST
166723              A:=:D:=A+C ; T+"LHPOI-LLPOI" ; * EXR ST
166727              A:=:D+16:=:D:=A+C
166733            FI
166733
166733          ELSE
166734
166734            IF A:=77 /\ T -FCLEAR=0 THEN
166740              A:=22 ; * BSTA 50 DA
166742              A=:SFCOM
166743            ELSE
166744
166744              T BZERO 14 =:CCBWO
166746              0=:OPWCH
166747              X:=XRG=:WCOUN
166751              A:=D=:BFDEV
166753              A:=0          ; T:=HDEV+"LHPOI" ; * EXR ST
166757              A:="CCBWO"+B ; T+"LLPOI-LHPOI" ; * EXR ST
166763              D:=0 BONE 10 BONE 1 ; * BSTA 50 DD
166767
166767              IF  D NBIT 5 THEN
166771
166771                A:=CCBWO SHZ 10 SHZ -16 =:X
166775
166775                IF A:=STEPR(X)><0 THEN D BONE 11 FI
167000                IF A:=DOORL(X)><0 THEN D BONE 12 FI
167003                IF A:=PRECP(X)><0 THEN D BONE 13 FI
167006                IF A:=DTRCK(X)><0 THEN D BONE 14 FI
167011                IF A:=COMFL   ><0 THEN D BONE 15 FI
167014
167014              FI
167014              A:=D=:SFCOM
```

```
%
 %    START NEW OPERATION
 %    SET K IF STREAMER
 %    START CREATING THE CONTROLLWORD IN D
 %
 %
 %    SHIFT TEST NO. IN PLACE
 %
 %    TAKE STERAMING FROM K
 %
 %
 %    GET IOX BITS, SAVE USE USER SPESIFYED CCB
 %
 %
 %
 %
 %
 %
 %
 %    EXEQUTE THE WANTED OUTPUT IOX
 %
 %
 %
 %
 %
 %
 %
 %    LOAD CCB POINTER LOW
 %    LOAD CCB POINTER HIG
 %
 %
 %
 %
 %
 %
 %    CLEAR + STREAMER IF STREAMER IN USE
 %    SAVE COMMAND
 %
 %
 %
 %
 %    NO. OF SECTORS TO TRANSFER
 %
 %    LOAD COMMAND POINTER, UPPER ADDR
 %    LOWER
 %    SET ACTVATE AND ENABELE INTERUPT BITS, BIT
 %
 %    IF NOT STREAMER
 %
 %    UNIT NO. TO X
 %
 %    SET BIT  9 IF STEP RATE
 %    SET BIT 10 IF DOOR LOCK
 %    SET BIT 11 IF DISABLE PRECOMP
 %    SET BIT 12 IF DUAL TRACK DENSITY (FUTURE!!
 %    SET BIT 13 IF HARDWARE COMPARE
 %
 %
 %    SAVE COMMAND
```

```
167016                AD:=MEMAD
167017              FI
167017
167017            FI
167017
167017            AD=:FMEMD
167020            T:=HDEV+LCONT
167022            A:=SFCOM ; * EXR ST
167024            A=:T
167025            IF A:=77/\TRG-FCLEAR=0 THEN
167031              T:=0; L+1
167033              A:=HDEV+RSTAT    ; * EXR SA
167036              A=:XRG
167037            FI
167037            A:=T
167040    BUSYR:  A=:BUSFL
167041            X:=XRG
167042            TAD:=TADRG
167043    RETUR:  EXITA
167044
167044    RBUS
167045
167045    *"
 167045
167045    @EOF
167045
```

```
%
%
%
%
%
%   MEMORY ADDRESS
%
%   ACTIVATE
%
%
%   FINISHED!!!
%   READ STATUS
%
%
%
%   BUSY
%
%
%
%
%
```

```
167045    ~
167045    %-----------------------------------------------------------------------
167045    %  PIT3-POF-CODE
167045    %
167045    %=======================================================================
167045    %
167045    *)KILL 3PITU 3POFU
167045    *'8UDMA+8VICO 3PIT
"167045    *"8PUDM+8PVIC -3PIT
"167045
167045    *"3PITU+3POFU
"167045
167045    *)KILL 3PITP 3POFP
167045    *"8PIOC 3PIT
"167045    *"8PPIO -3PIT
"167045    *"3PITP+3POFP
"167045
167045    *"3PIT
"167045
167045    *"-3PIT
"167045    * 3PIT; *-1/        % MARK THAT NEXT TIME WE ARE ON PIT 3
167045    *'
"167045    @EOF
167045

167045   "8BDIS+8BDIM+8ZBDI
```

```
167045
167045
167045  %=============================================================================
167045  % 43.5        B D I S K                    VERSION A       26/11-81    CORR. BY TP
167045  %                                         VERSION B       30/12-81    CORR. BY TP
167045  %                                         VERSION C.6     16/ 4-82    CORR. BY TP
167045  %                                         VERSION D        6/10-82    CORR. BY TP
167045  %                                         VERSION E       10/11-82    CORR. BY TP
167045  %                                         VERSION E.1     29/ 3-83    CORR. BY TP
167045  %                                         VERSION E.2     31/ 8-83    CORR. BY TP
167045  %                                         VERSION E.3     26/ 9-83    CORR. BY TP
167045  %                                         VERSION E.4     19/ 3-84    CORR. BY TP
167045  %                                         VERSION F.0      2/ 4-84    CORR. BY TP
167045  %                                         VERSION F.1      3/ 5-84    CORR. BY TP
167045  %                                         VERSION F.2     26/ 6-84    CORR. BY TP
167045  %                                         VERSION F.3     31/ 7-84    CORR. BY TP
167045  %                                         VERSION F.4      3/ 9-84    CORR. BY TP
167045  %
167045  %   SUPER DISK (AND ECC CARTRIDGE DISKS) TRANSFER ROUTINE
167045  %
167045  %           THERE ARE 3 VERSIONS OF THE DRIVER, ACCORDING TO LIBRARY MARKS:
167045  %
167045  %                           8BDIS = NORMAL SINTRAN DRIVER
167045  %                           8BDIM = TEST PROGRAM DRIVER
167045  %                           8ZBDI = SWAP DRIVER (READ ONLY)
167045  %
167045  %       8BDIS AND 8BDIM
167045  %
167045  %           THE ARRAY ',B HTABL' MUST BE CORRECT (SEE BELOW).
167045  %
167045  %       8BDIS
167045  %
167045  %           THE DISC-DESCRIBING TABLES ARE EXTERNAL (SEE BELOW).
167045  %
167045  %       8ZBDI
167045  %
167045  %           THE CORRECT DEVICE DESCRIPTOR IS GENERATED AT ASSEMBLY TIME!!!
167045  %           PLDWO AND PSTWO WILL NOT WORK FOR N10.
167045  %
167045  %       FOR ALL:
167045  %
167045  %           IF THE ROUTINES 'PLDWO' AND 'PSTWO' ARE UNDEFINED, THEY WILL
167045  %           BE INCLUDED AFTER THE DRIVER!!!
167045  %           FOR N10, PAGING MUST BE OFF WHEN PLDWO AND PSTWO ARE ENTERED
167045  %           (USED BY ERROR CORRECTION). NOT FOR 8ZBDI. SEE 8ZBDI ABOVE.
167045  %           ERROR CORRECTION ASSUMES THAT THE CPU CAN REACH ALL THE MEMORY
167045  %           ACCESSIBLE TO THE DISK.
167045  %
167045  %
167045  '8BDIS+8BDIM
167045  %
167045  %       CALL:
167045  %       JPL  I (BDISK
167045  %       JMP    ERROR            %ERROR EXIT
167045  %       JMP    BUSY             % BUSY EXIT
167045  %       JMP    FINIS            %FINISHED EXIT
167045  %
167045  '8ZBDI+8BDIS+8BDIM
167045  %
167045  %       REGISTER CONTENTS WHEN ROUTINE IS CALLED
```

```
167045  %
167045  %       T-  BIT 0-5:  DEVICE OPERATION  (FOR 8ZBDI, READ ONLY)
167045  %                      0 READ TRANSFER
167045  %                      1 WRITE TRANSFER
167045  %                      2 READ PARITY TRANSFER
167045  %                      3 COMPARE TRANSFER
167045  %                     20 READ LAST STATUS
167045  %                     35 RELEASE                (NOT FOR THE SWAP DRIVER)
167045  %                     36 PRIORITY-SELECT        (NOT FOR THE SWAP DRIVER)
167045  %                     37 THIS FUNCTION BELONGS TO FUNCTION 42 AND IS STARTED
167045  %                        BY THE DRIVER. IT SHOULD NOT BE USED IN A CALL.
167045  %                     42 TEST IF SPARE TRACKS AND IF SPARE SECTOR FORMAT.
167045  %                        (NOT FOR THE SWAP DRIVER).
167045  %                        IT IS ASSUMED THAT NO OF SECTORS IN X IS 1 AND THAT
167045  %                        SECTOR NUMBER (LOGICAL DISK ADDR) IN AD IS 0 (EXCEPT
167045  %                        THE FIXED BIT IN D 017) WHEN THIS FUNCTION CALLS THE
167045  %                        DRIVER THE FIRST TIME.
167045  %                     43 READ  FORMAT TABLE (NOT FOR THE SWAP DRIVER)
167045  %                     44 WRITE FORMAT TABLE (NOT FOR THE SWAP DRIVER)
167045  %           BIT 6-8:  UNIT SELECT
167045  %           BIT 9-11:         SURFACE NUMBER FOR PHOENIX DISK
167045  %           BIT 12-13: CORE ADDRESS BITS 16-17 (N10)
167045  %           BIT 14:   DEVICE TYPE
167045  %                         1= 30/60/90 MBYTE
167045  %                         0= ANY OTHER TYPE
167045  %
167045  %           BIT 15:   0: NORMAL TRANSFER
167045  %                     1: BIT 6 IS ADDED TO THE SECTOR NUMBER
167045  %                        (USED TO READ SPARE TRACKS. ONLY POSSIBLE FOR
167045  %                        THE LIBRARY MARK 8BDIM).
167045  %
167045  %       A- LOGICAL ADDRESS (DISK SECTOR NUMBER ) BITS 16-31
167045  %
167045  %       D- LOGICAL ADDRESS (DISK SECTOR NUMBER ) BITS 0-15
167045  %                         LOGICAL ADDRESS BIT 15 MEANS ANY FIXED
167045  %                         SURFACE FOR CARTRIDGE DISK
167045  %
167045  %       X- NUMBER OF SECTORS TO BE TRANSFERED
167045  %
167045  %          ****   ,B MEMAD CONTAINS THE 24-BIT MEMORY ADRESS
167045  %                        IN A DOUBLE WORD.
167045  %          ****   ,B HDEV  CONTAINS HARDWARE DEVICE NUMBER
167045  %
167045  "8BDIS+8BDIM
167045  %
167045  %          ****   ,B HTABL IS A 4 ENTRY INTEGER ARRAY OF POINTERS TO
167045  %                        DISC ADRESS LAYOUT FOR UNITS 0,1,2 AND 3.
167045  %
167045  "8ZBDI+8BDIS+8BDIM
167045  %
167045  %        DISC LAYOUT DEFINITION:
167045  %
167045  %                                WORD 0 (BSECW) = WORDS/SECTOR
167045  %                                     1 (BSECT) = SECTORS/TRACK
167045  %                                     2 (BSECY) = SECTORS/CYLINDER
167045  %                                     3 (BMXCY) = MAXIMUM VALUE OF CYLINDER
167045  %                                     4           FIRST CYL OF SPARE TRACK AREA
167045  %                                     5           FORMAT IN ECC CONTROL WORD
167045  %                                     6           FIRST CYL OF EXTRA AREA
167045  %                                     7           POINTER TO RELATED LAYOUT
```

```
167045  %
167045  %     EXIT INFORMATION:
167045  %
167045  %     ERROR EXIT: X- HARDWARE STATUS REG.
167045  %                 T- BIT  9: SEEK ERROR. RETURN TO ZERO SEEK HAS BEEN DONE.
167045  %                    BIT 10: CORE ADDR. REG NOT AS EXPECTED
167045  %                    BIT 11: LOG. BLOCK ADDR. OUT OF RANGE, OR
167045  %                            TRANSFER BIGGER THAN 63.5K
167045  %                    BIT 12: DATA CORRECTION INFORMATION ERROR
167045  %                    BIT 13: DRIVE TYPE DEFINITION ERROR
167045  %                    BIT 14: ILL. DEVICE OP. CODE
167045  %                    BIT 15: CONTROLLER NOT ACTIVE AFTER ACTIVATE
167045  %     EVERY TIME THE DRIVER ERROR EXITS, IT STORES 11 WORDS AT THE END OF
167045  %     THE CODE. THESE WORDS, THEREFORE, WILL CONTAIN SIGNIFICANT INFORMATION
167045  %     ABOUT THE LAST ERROR THAT OCCURED. THE ADDRESS OF THIS FIELD CAN BE
167045  %     FOUND IN BDISK+2.
167045  %
167045  %     BUSY EXIT:      THE ROUTINE MUST BE CALLED AGAIN AT ONCE, OR
167045  %                     AFTER INTERRUPT, THE NEW CALL IS DONE WITH T,A,D AND
167045  %                     X-REG. UNCHANGED.
167045  %
167045  %     FINISHED EXIT: X-HARDWARE STATUS REG.
167045  %                    WITH FUNCTION 042, X WILL BE:
167045  %                    0: NOT SPARE TRACKS, NOT SPARE SECTOR FORMAT
167045  %                    1:     SPARE TRACKS, NOT SPARE SECTOR FORMAT
167045  %                    2: NOT SPARE TRACKS,     SPARE SECTOR FORMAT
167045  %                    3:     SPARE TRACKS,     SPARE SECTOR FORMAT
167045  %
167045  %     B-REG. IS NOT CHANGED BY THE ROUTINE
167045  %
167045  %       THE ROUTINE CONVERTS LOGICAL (SECTOR NUMBER) ADDR. (LA)
167045  %       TO HARDWARE (CYLINDER/HEAD/SECTOR) ADDR. (HA)
167045  %       ACCORDING TO TABLE HTABL FOR THE CORRESPONDING UNIT.
167045  %
167045  %     **** THE NUMBER OF SECTORS FOR THE VARIOUS DISK PACKS, AND THE ACTUAL TABLE NAMES:
167045  %
167045  %          37 MB =  110176                      : DT037
167045  %          70 MB =  220526                      : DT070
167045  %          75 MB =  220526                      : DT075
167045  %         135 MB =  421206                      : DT135   SPARE SECTOR FORMAT
167045  %         140 MB =  441254                      : DT140
167045  %         160 MB =  441254                      : DT160
167045  %         288 MB = 1045572                      : DT288
167045  %         285 MB = 1066450                      : DT285   SPARE SECTOR FORMAT
167045  %         300 MB = 1127720                      : DT300
167045  %         450 MB = 1542420                      : DT450   SPARE SECTOR FORMAT
167045  %         460 MB = 1603760                      : DT460
167045  %
167045  %          30 MB = 0034736 FOR EACH SURFACE     : DT030
167045  %          60 MB = 0034736 FOR EACH SURFACE     : DT030
167045  %          90 MB = 0034736 FOR EACH SURFACE     : DT030
167045  %
167045  %
167045  %       WHEN CALLING, THE LOGICAL ADDR. IN AD MUST
167045  %       BE REPRESENTED AS AN UNSIGNED MAGNITUDE NUMBER.
167045  %
167045  %
167045  %       THE SECTORS WITHIN THE LAST CYLINDER ARE TRANSFERED
167045  %       FIRST, THEN THE SECTORS WITHIN THE PREVIOUS CYLINDER ETC....
167045  %       FOR THE OLD INTERFACE (10 KHZ), WITHIN ONE CYLINDER,
```

```
167045  %         THE GREATEST SINGLE TRANSFER IS  64K-BSECW, WHERE BSECW IS NUMBER
167045  %         OF WORDS PER SECTOR (THE WORD COUNT REGISTER HAS ONLY 16 BITS).
167045  %         FOR THE NEW INTERFACE (15KHZ), STILL WITHIN ONE CYLINDER,
167045  %         THE GREATEST SINGLE TRANSFER CAN BE A WHOLE CYLINDER, SINCE
167045  %         THE WORD COUNT REGISTER HAS 24 BITS.
167045  %
167045  %      DISPLACEMENTS FOR IOX INSTRUCTIONS:
167045  %
167045           RCA=0                  % READ CORE ADDRESS
167045           LCA=1                  % LOAD CORE ADDRESS
167045           RSC=2                  % READ SEEK CONDITION / ECC COUNT
167045           LBA=3                  % LOAD BLOCK ADDRESS  I/II
167045           RSR=4                  % READ STATUS REG / ECC PATTERN
167045           LCO=5                  % LOAD CONTROL WORD
167045           LWC=7                  % LOAD WORD COUNT / ECC CONTROL
167045         INIOX=165540             % INITIAL IOX INSTRUCTION
167045
167045  % THE DISK DRIVER MAY BE CALLED WITH 11 DIFFERENT DEVICE OPERATIONS.
167045  % IN THE FOLLOWING, READ, READ PARITY, AND COMPARE, ARE CALLED NON-WRITE.
167045  % THE WRITE OPERATION IS CALLED WRITE. THE READ-STATUS OPERATION IS QUITE
167045  % SPECIAL. IT IS REALLY A READ-LAST-STATUS OPERATION, AS IT WILL READ
167045  % LAST STATUS FROM THE VARIABLE SSTAT. IT WILL ALSO INITIATE THE DISK DATA
167045  % FIELD VARIABLES BUSFL, ECCFL, MARGC, AND SMARG BY STORING ZERO INTO THEM,
167045  % AND BY SETTING ERRC1 TO -4 AND ERRC2 TO -28.
167045  % THE TEST-IF-SPARE-TRACK OPERATION WILL BE EXECUTED FIRST AS A PARITY READ
167045  % FROM THE LAST POSSIBLE SECTOR ON THE FIRST TRACK, AND THEN AS A PARITY READ
167045  % FROM THE FIRST SECTOR IN THE SPARE TRACK AREA. IT IS NOT FOR THE SWAP DRIVER.
167045  % RELEASE AND PRIORITY-SELECT ARE NOT FOR THE SWAP DRIVER EITHER. THEY ARE
167045  % USED WHEN MORE THAN ONE CPU USE THE DISK.
167045  %
167045  % ***** THE READ-STATUS OPERATION SHOULD NOT BE USED WHEN BUSFL IS NONZERO.****
167045  %
167045  % IF THE DISK DRIVER IS CALLED WITH READ-ZERO-SECTORS, THE STATUS REGISTER
167045  % WILL BE READ AND PUT IN THE X REGISTER, AND THE DISK DRIVER WILL EXIT
167045  % AT THE FINISHED EXIT. IF NOT-ON-CYLINDER (STATUS BIT 14),
167045  % THE DISK DRIVER WILL EXIT AT THE ERROR EXIT.
167045  %
167045  %
167045  % THE VARIABLES IN THE DISK DATA FIELD USED BY THE DISK DRIVER:
167045  %
167045  % ARG
167045  %
167045  % USED TO SAVE THE VALUE OF THE A-REGISTER AT ENTRY.
167045  %
167045  % BUSFL
167045  %
167045  % BUSY FLAG. CAN HAVE THE VALUES ZERO OR NONZERO.
167045  % MUST BE ZERO THE FIRST TIME THE DISK DRIVER IS CALLED. THE DISK DRIVER
167045  % WILL SET BUSFL TO NONZERO WHEN A TRANSFER IS STARTED, AND WILL SET IT BACK
167045  % TO ZERO WHEN A TRANSFER IS FINISHED, OR WHEN IT ERROR EXITS.
167045  %
167045  % CDISP
167045  %
167045  % USED FOR ECC COUNT AND SECTOR WORD NUMBER.
167045  % WHEN THE DISK DRIVER TRIES TO DO ERROR CORRECTION ON A SECTOR, IT READS
167045  % THE ECC COUNT AND STORES IT IN CDISP. IF 1 < CDISP < 8204, IT IS CHANGED
167045  % INTO A SECTOR WORD NUMBER IN THE RANGE 0-0777, POINTING TO A DOUBLE WORD
167045  % IN THE SECTOR THAT MAY BE CHANGED, DEPENDING ON THE ECC PATTERN (SEE
167045  % CPAT1 AND CPAT2). IF ERROR CORRECTION HAS TAKEN PLACE (SEE CORCU), THEN
167045  % IT IS POSSIBLE FOR THE CALLING PROGRAM TO KNOW WHERE THE CORRECTION TOOK
```

```
167045  % PLACE, BY INSPECTING CDISP.
167045  %
167045  % CMAD1
167045  %
167045  % POINTS TO THE FIRST WORD OF A DOUBLE WORD CONTAINING CURRENT MEMORY ADDRESS.
167045  % THE CURRENT MEMORY ADDRESS IS THE MEMORY ADDRESS USED BY THE DRIVER
167045  % WHEN IT STARTS A TRANSFER. IT IS NOT NECESSARILY EQUAL TO THE MEMORY
167045  % ADDRESS SUPPLIED IN THE CALL (SEE MEMA1, MEMA2).
167045  % IF THE DISK DRIVER BREAKS A TRANSFER INTO SMALLER TRANSFERS, FOR INSTANCE,
167045  % IT HAS TO CALCULATE A NEW MEMORY ADDRESS AND STORE IT IN CMAD1 AND CMAD2.
167045  %
167045  % CMAD2
167045  %
167045  % POINTS TO THE SECOND WORD OF A DOUBLE MEMORY ADDRESS. SEE CMAD1.
167045  %
167045  % CORCU
167045  %
167045  % CORRECTION COUNTER. EVERY TIME THAT ERROR CORRECTION OCCURS,
167045  % CORCU IS INCREMENTED BY 1. ERROR CORRECTION IS ALWAYS DONE ON SINGLE SECTORS.
167045  % THE NOT-WRITE RETRY COUNTER (SEE SRTRY) IS USUALLY ALSO INCREMENTED (ONCE
167045  % FOR EACH SECTOR IN THE TRANSFER) WHEN CORCU IS INCREMENTED.
167045  % CORCU IS INCREMENTED ON READ AND PARITY-CHECK OPERATIONS.
167045  % THE FAILING BITS CAN BE IN THE DATA PART AND/OR IN THE ECC POLYNOMIAL.
167045  %
167045  % CPAT1
167045  %
167045  % WHEN THE DISK DRIVER WANTS TO DO ERROR CORRECTION, IT READS THE ELEVEN BITS  ...
167045  % OF THE ECC PATTERN, INVERTS THE BIT ORDER (SWAPS BIT 0 AND 10, BIT 1 AND 9,
167045  % ETC.), AND STORES IT TEMPORARILY IN CPAT1. CPAT1 SHOULD NOT BE ZERO.
167045  % THEN CPAT1 IS SHIFTED ACCORDING TO THE LEAST SIGNIFICANT BITS OF THE ECC
167045  % COUNT IN CDISP. THE RESULT IS STORED IN CPAT1 AND CPAT2, READY TO BE
167045  % EXCLUSIVE OR'ED INTO THE SECTOR JUST READ.
167045  %
167045  % CPAT2
167045  %
167045  % SEE CPAT1.
167045  %
167045  % DRG
167045  %
167045  % USED TO SAVE THE VALUE OF THE D-REGISTER AT ENTRY.
167045  %
167045  % ECCFL
167045  %
167045  % RUN-ECC OPERATION FLAG. CAN HAVE THE VALUES ZERO AND NONZERO.
167045  % THIS FLAG IS USUALLY ZERO, BUT WHEN THE DISK DRIVER WANTS TO DO ERROR
167045  % CORRECTION, IT SETS ECCFL TO NONZERO AND STARTS THE RUN-ECC OPERATION.
167045  % WHEN RUN-ECC IS FINISHED, ECCFL IS SET BACK TO ZERO.
167045  % ECCFL IS SET TO ZERO AT THE ERROR EXIT AND THE FINISHED EXIT.
167045  %
167045  % ERRC1
167045  %
167045  % ERROR COUNTER. SHOULD HAVE A NEGATIVE VALUE. BEFORE THE DISK DRIVER IS CALLED
167045  % THE FIRST TIME, ERRC1 USUALLY SHOULD HAVE THE VALUE -4. A FAILING WRITE
167045  % TRANSFER WILL THEN BE RETRIED THREE TIMES. IF THE CALLING PROGRAM WANTS
167045  % NOT TO HAVE WRITE RETRIES, IT SHOULD STORE -1 IN ERRC1 BEFORE CALLING
167045  % THE DISK DRIVER. FOR FAILING NOT-WRITE TRANSFERS, ERRC1 SHOULD BE HANDLED
167045  % IN THE SAME WAY, BUT THEN ERRC2 MUST ALSO BE TAKEN INTO ACCOUNT.
167045  % ERRC1 IS SET TO -4 WHEN A TRANSFER IS FINISHED, WHEN THE DISK DRIVER IS
167045  % IN ERROR-RECOVERY MODE (SEE SMARG), WHEN ALL WRITE-RETRIES HAVE BEEN DONE,
167045  % OR WHEN ALL NOT-WRITE MARGINAL RECOVERIES HAVE BEEN DONE (SEE MARGC).
```

```
167045  %
167045  % ERRC2
167045  %
167045  % MARGINAL RECOVERY COUNTER. SHOULD HAVE A NEGATIVE VALUE. BEFORE THE DISK
167045  % DRIVER IS CALLED THE FIRST TIME, ERRC2 USUALLY SHOULD HAVE THE VALUE -28.
167045  % A FAILING NOT-WRITE TRANSFER WILL FIRST DO SOME RETRIES ACCORDING TO THE
167045  % VALUE OF ERRC1. WHEN ERRC1 HAS BEEN COUNTED UP TO ZERO, THERE WILL BE NO
167045  % MORE ORDINARY RETRIES. INSTEAD, THE DISK DRIVER WILL TRY TO DO MARGINAL
167045  % RECOVERY CYCLES, ACCORDING TO THE VALUE OF ERRC2. IT WILL FIRST SET MARGC
167045  % NONZERO. USUALLY, THE DISK DRIVER WILL DO 27 MARGINAL RECOVERIES (THREE
167045  % FULL CYCLES). BUT IF THE CALLING PROGRAM SETS BOTH ERRC1 AND ERRC2 TO -1,
167045  % THE DISK DRIVER WILL DO NO ORDINARY RETRIES AND NO MARGINAL RECOVERY CYCLES.
167045  % ERRC2 IS SET TO -28 WHEN A TRANSFER IS FINISHED, WHEN THE DISK DRIVER IS IN
167045  % ERROR-RECOVERY MODE (SEE SMARG), WHEN ALL WRITE-RETRIES HAVE BEEN DONE,
167045  % OR WHEN ALL NOT-WRITE MARGINAL RECOVERIES HAVE BEEN DONE (SEE MARGC).
167045  %
167045  % HDEV
167045  %
167045  % CONTAINS HARDWARE DEVICE NUMBER (01540 OR 01550).
167045  %
167045  % HTABL
167045  %
167045  % THERE ARE FOUR WORDS IN THE DATAFIELD (ONE FOR EACH UNIT), EACH POINTING TO
167045  % AN EIGHT-WORD DISK LAYOUT VECTOR. HTABL POINTS TO THE FIRST OF THE FOUR DATA
167045  % FIELD WORDS.
167045  %
167045  % MARGC
167045  %
167045  % MARGINAL RECOVERY FLAG. CAN HAVE THE VALUES ZERO AND NONZERO.
167045  % WHEN ALL THE ORDINARY NOT-WRITE RETRIES HAVE BEEN DONE (IF A TRANSFER FAILS),
167045  % THE MARGINAL RECOVERY BIT IN THE CONTROL WORD IS TURNED ON, MARGC IS SET
167045  % TO NONZERO, AND TYPEC IS INCREMENTED IN BIT 9.
167045  % MARGC IS SET TO ZERO WHEN THE DISK DRIVER IS IN ERROR RECOVERY
167045  % MODE (SEE SMARG), AND WHEN A TRANSFER IS FINISHED (BOTH OK EXIT AND ERROR
167045  % EXIT).
167045  %
167045  % MEMA1
167045  %
167045  % THE MOST SIGNIFICANT PART OF A DOUBLE WORD CONTAINING MEMORY ADDRESS.
167045  %
167045  % MEMA2
167045  %
167045  % THE LEAST SIGNIFICANT PART OF THE MEMORY ADDRESS (SEE MEMA1).
167045  %
167045  % NWLBA
167045  %
167045  % NEW SURFACE AND SECTOR. USED WHEN SPARE TRACK ALLOCATION IS ACTIVE.
167045  % A TRACK THAT IS FLAGGED AS A BAD TRACK, CONTAINS THE ADDRESS OF THE
167045  % RESERVE TRACK IN EVERY DOUBLE WORD. THE FIRST PART OF THE DOUBLE WORD
167045  % CONTAINS THE NEW CYLINDER NUMBER, AND THE SECOND PART CONTAINS THE NEW
167045  % SURFACE NUMBER IN THE LEFTMOST 8 BITS, AND THE SECTOR NUMBER IN THE
167045  % RIGHTMOST 8 BITS. THIS SECTOR NUMBER IS IRRELEVANT, AS THE OLD SECTOR
167045  % NUMBER (THE SECTOR NUMBER IN THE ORIGINAL DISK TRANSFER) IS USED INSTEAD.
167045  % NWLBA MUST BE NWLBB+1. (*****)
167045  %
167045  % NWLBB
167045  %
167045  % NEW CYLINDER NUMBER. SEE NWLBA. MUST BE NWLBA-1 !!
167045  %
167045  % QCMD1
```

```
167045  %
167045  % OLD COMPUTED MEMORY ADDRESS, BITS 23-16. USED TO SAVE THE MEMORY ADDRESS
167045  % OF A DISK TRANSFER DURING SPARE TRACK ALLOCATION.
167045  %
167045  % OCMD2
167045  %
167045  % OLD COMPUTED MEMORY ADDRESS, BITS 15-0. SEE OCMD1.
167045  %
167045  % OSVBA
167045  %
167045  % OLD SURFACE AND SECTOR. USED TO SAVE THE OLD VALUES OF THE DISK ADDRESS
167045  % DURING SPARE TRACK ALLOCATION.
167045  %
167045  % OSVBB
167045  %
167045  % OLD CYLINDER. SEE OSVBA.
167045  %
167045  % OSVCO
167045  %
167045  % OLD CONTROL WORD. USED TO SAVE THE OLD CONTROL WORD DURING SPARE TRACK
167045  % ALLOCATION.
167045  %
167045  % OSVWC
167045  %
167045  % OLD WORD COUNT, LEAST SIGNIFICANT 16 BITS.
167045  % USED TO SAVE THE OLD WORD COUNT DURING SPARE TRACK ALLOCATION.
167045  %
167045  % OSVWK
167045  %
167045  % OLD WORD COUNT, MOST SIGNIFICANT 8 BITS.
167045  % USED TO SAVE THE OLD WORD COUNT DURING SPARE TRACK ALLOCATION.
167045  %
167045  % SCADR
167045  %
167045  % THE LEAST SIGNIFICANT PART OF THE EXPECTED CORE ADDRESS REGISTER WHEN
167045  % A TRANSFER HAS FINISHED. IT IS COMPUTED BY THE DISK DRIVER (CMAD2+SVLWC).
167045  %
167045  % SLONG
167045  %
167045  % COUNTER FOR ERROR-RECOVERY (SEE SMARG). WHEN THE DISK DRIVER INITIATES
167045  % ERROR-RECOVERY MODE, SLONG IS SET TO 1-N, WHERE N IS THE NUMBER OF SECTORS
167045  % IN THE TRANSFER.
167045  %
167045  % SMARG
167045  %
167045  % ERROR-RECOVERY FLAG. CAN HAVE THE VALUES ZERO OR NONZERO.
167045  % WHEN A NOT-WRITE TRANSFER FAILS, THE DISK DRIVER INITIATES ERROR-RECOVERY
167045  % BY SETTING SMARG TO NONZERO. ERROR-RECOVERY MEANS THAT THE TRANSFER IS
167045  % BROKEN INTO SINGLE-SECTOR TRANSFERS. EACH OF THESE TRANSFERS IS THEN DONE
167045  % AND CHECKED FOR ERRORS. IF THE TRANSFER IS READ, AND IT FAILS, ERROR
167045  % CORRECTION IS TRIED. IF THIS FAILS, OR IF THE TRANSFER IS PARITY CHECK OR
167045  % COMPARE, THE USUAL RETRIES AND MARGINAL RECOVERIES ARE DONE.
167045  % SMARG IS SET TO ZERO WHEN A NEW TRANSFER IS STARTED, AND WHEN A TRANSFER
167045  % IS FINISHED (BOTH OK EXIT AND ERROR EXIT).
167045  %
167045  % SPACO
167045  %
167045  % SPARE TRACK ALLOCATION COUNTER. EVERY TIME A DISK TRANSFER HAS SUCCESSFULLY
167045  % USED A RESERVE TRACK, THIS COUNTER IS INCREMENTED BY ONE.
167045  %
```

```
167045  % SPAFL
167045  %
167045  % SPARE TRACK ALLOCATION FLAG.
167045  % 0:  NORMAL TRANSFER (NO SPARE TRACKS INVOLVED).
167045  % 1:  ADDRESS MISMATCH HAS OCCURED. SPARE TRACK ADDRESS-READ HAS BEEN STARTED.
167045  % 2:  TRANSFER IN THE SPARE TRACK AREA HAS BEEN STARTED.
167045  %
167045  % SRTRY
167045  %
167045  % NOT-WRITE RETRY COUNTER. IT IS INCREMENTED EVERY TIME AN ORDINARY NOT-WRITE
167045  % RETRY IS STARTED. MARGINAL RECOVERY CYCLE RETRIES ARE COUNTED IN TYPEC,
167045  % BITS 15-9, EVERY TIME MARGC IS SET NONZERO.
167045  %
167045  % SVLBA
167045  %
167045  % SAVED DISK ADDRESS (SURFACE AND SECTOR).
167045  %
167045  % SVLBB
167045  %
167045  % SAVED DISK ADDRESS (CYLINDER NUMBER).
167045  %
167045  % SVLCA
167045  %
167045  % SAVED LEAST SIGNIFICANT PART OF THE CURRENT MEMORY ADDRESS (EQUAL TO CMAD2).
167045  %
167045  % SVLCO
167045  %
167045  % SAVED CONTROL WORD, EXCEPT WHEN RUN-ECC OPERATION IS STARTED.
167045  %
167045  % SVLWC
167045  %
167045  % SAVED WORD COUNT, LEAST SIGNIFICANT 16 BITS (NUMBER OF SECTORS * BSECW).
167045  %
167045  % SVLWK
167045  %
167045  % SAVED WORD COUNT, MOST SIGNIFICANT 8 BITS (NUMBER OF SECTORS * BSECW).
167045  %
167045  % SWTRY
167045  %
167045  % WRITE RETRY COUNTER. SWTRY IS INCREMENTED BY ONE EACH TIME A WRITE TRANSFER
167045  % FAILS.
167045  %
167045  % TRG
167045  %
167045  % USED TO SAVE THE VALUE OF THE T-REGISTER AT ENTRY.
167045  %
167045  % TYPEC
167045  %
167045  % BIT    0:    1: ECC INTERFACE                    0: 33/66MB INTERFACE
167045  %       1:    1: HEAD ADVANCE                        0: NOT HEAD ADVANCE
167045  %       2:    1: ERROR CORRECTION NOT ALLOWED        0: ERROR CORRECTION ALLOWED
167045  %       3:    1: MARGINAL RECOVERY NOT ALLOWED       0: MARGINAL RECOVERY ALLOWED
167045  %       4:    1: HEAD ADVANCE NOT ALLOWED, UNIT 0    0: HEAD ADVANCE ALLOWED, UNIT 0
167045  %             (SPARE TRACK ALLOC. ALLOWED)              (NO SPARE TRACK ALLOCATION)
167045  %       5:    1: DITTO FOR UNIT 1                    0: DITTO FOR UNIT 1
167045  %       6:    1: DITTO FOR UNIT 2                    0: DITTO FOR UNIT 2
167045  %       7:    1: DITTO FOR UNIT 3                    0: DITTO FOR UNIT 3
167045  %       8:    1: AUTOMATIC RELEASE NOT ALLOWED       0: ALLOWED
167045  %       9:    1: NEW (15 KHZ) INTERFACE              0: OLD (10 KHZ) INTERFACE
167045  %   15-10:    MARG. REC. COUNTER (INCREMENTED EACH TIME MARGC IS SET NONZERO)
```

```
167045  %
167045  %     BITS 0 AND 1 ARE SET INSIDE THE DISK DRIVER.
167045  %     BITS 2, 3, AND 4-7 ARE SET BY THE CALLING PROGRAM.
167045  %             BITS 4-7 CAN ALSO BE SET BY THE FUNCTION TEST-IF-SPARE-TRACK (042)
167045  %     BIT  8 IS SET BY THE FUNCTION PRIORITY-SELECT, AND RESET BY THE FUNCTION
167045  %             RELEASE. IT MAY ALSO BE SET/RESET BY THE CALLING PROGRAM.
167045  %     BIT  9 IS SET INSIDE THE DISK DRIVER.
167045  %
167045  % XRG
167045  %
167045  % USED TO SAVE THE VALUE OF THE X-REGISTER AT ENTRY.
167045  %
167045
167045  "8BDIS+8BDIM
167045
167045  BDISK=*
167045          JMP    MORE9
167046          #F4                     % VERSION LETTER AND NUMBER
167047          DRIAR                   % ADDRESS OF ERROR FIELD
167050
167050  BSECW,  1000                    % WORDS/SECTOR
167051  BSECT,    22                    % SECTORS/TRACK
167052  BSECY,   132                    % SECTORS/CYLINDER
167053  BMXCY,  1466                    % MAX CYLINDER VALUE
167054          1465                    % FIRST CYL NO OF THE SPARE TRACK AREA
167055          0                       % FORMAT (0 OR 020)
167056          0                       % FIRST CYLINDER OF EXTRA AREA
167057          0                       % POINTER TO RELATED LAYOUT
167060
167060  "8BDIS+8BDIM
167060
167060  MORE9,  STF    TRG ,B
167061          STX    XRG ,B
167062          SHT    SHR 6
167063          SAX    7                % MASK FOR UNIT
167064          RAND   ST DX            % X = UNIT NO.
167065          AAX    HTABL
167066          RADD   SB DX
167067          LDX    ,X               % X => DISC DEFINITION
167070          LDF    ,X 0             % GET DEFINITION
167071          STF    BSECW            % TO CURRENT COPY
167072          LDF    ,X 3
167073          STF    BSECW+3
167074          LDD    6 ,X
167075          STD    BSECW+6
167076          LDF    TRG ,B
167077          LDX    XRG ,B
167100          LDA    HDEV,B           % HARDWARE DEVICE NUMBER
167101          ADD    (IOX  LCO        % CONSTRUCT FIRST IOX
167102          SUB I  (BLCOG           % TEST FIRST IOX IF INITIALIZED
167103          JAF    NOIOX            % BRANCH IF NOT INITIALIZED
167104          ADD I  (BLCOG
167105          SUB I  (BLCO4           % TEST LAST IOX
167106          JAZ    BRCHK            % JUMP IF INITIALIZED
167107  NOIOX,  ADD I  (BLCO4           % INITIALIZE ALL IOX
167110          AAA    -5
167111          STA I  (BRCA
167112          STA I  (BRCA2
167113          AAA    2
167114          STA I  (BRSC1
```

```
167115          STA I (BRSC2
167116          STA I (BRSC3
167117          STA I (BRSC4
167120          STA I (BRSC5
167121          STA I (BRSC6
167122          STA I (BRSC7
167123          STA I (BRSC8
167124          AAA   2
167125          STA I (BRSR1
167126          STA I (BRSR2
167127          STA I (BRSR3
167130          STA I (BRSR4
167131          STA I (BRSR5
167132          STA I (BRSR6
167133          STA I (BRSR8
167134          AAA   -3
167135          STA I (BLCA1
167136          STA I (BLCA3
167137          AAA   2
167140          STA I (BLBA1
167141          STA I (BLBA3
167142          AAA   2
167143          STA I (BLCO1
167144          STA I (BLCO3
167145          STA I (BLCO4
167146          STA I (BLCO8
167147          STA I (BLCOF
167150          STA I (BLCOG
167151          STA I (BLCOH
167152          AAA   2
167153          STA I (BLWC1
167154          STA I (BLWC2
167155          STA I (BLWC3
167156
167156  "8ZBDI+8BDIS+8BDIM
167156
167156          JMP   BRCHK
167157
167157  )FILL
167217
167217  BRCHK,  COPY  SL DA             % SAVE L
167220          STA I (9LREG
167221          SAA   77
167222          AND   TRG ,B
167223
167223  "8BDIS+8BDIM
167223
167223          AAA   -042
167224          JAZ   L41               % ZERO: TEST-IF-SPARE-TRACK
167225          AAA   042-043
167226          JAZ   L41               % ZERO: READ-FORMAT-TABLE
167227          AAA   043-044
167230          JAZ   L41               % ZERO: WRITE-FORMAT-TABLE
167231          AAA   044-037
167232          JAZ   L41               % ZERO: PART OF TEST-IF-SPARE-TRACK
167233          AAA   037-035
167234          JAZ   L47               % ZERO: RELEASE
167235          AAA   035-036
167236          JAF   L48
167237          JMP I (L49              % PRIORITY-SELECT
```

```
167240  L47,    JMP I (L50
167241  L48,    AAA   036-020
167242          JAF   L25
167243          LDA   BUSFL ,B          % OPERATION IS READ LAST STATUS
167244          JAF   L39               % NONZERO: BUSY. DONT READ LAST STATUS
167245          JMP I (FINEX
167246  L25,    AAA   14
167247          SKP   IF DA LST 0
167250
167250  "8BDIS+8BDIM+8ZBDI
167250
167250          JMP I (ILCOD            % ILL. DEVICE OP. CODE
167251  L41,    LDA   TRG ,B
167252          AND   (0700             % MASK FOR UNIT ADDRESS BITS.
167253          SHA   ZIN  1            % SHIFT TO POSITION FOR CWR.
167254          STA I (BTSTA            % STORE VALUE
167255          LDA   SVLCO,B           % GET LAST COMMAND
167256          AND   (16000            % MASK FOR WRITE AND COMPARE BIT, AND MARG.REC. BIT
167257          ADD I (BTSTA            % ADD TO NEW COMMAND IN ORDER NOT TO
167260          STA I (BTSTA            % REVERSE THE FIFO DIRECTION IN BLCOG
167261          LDA   BUSFL,B           % PREVIOUS X-FER TO BE CHECKED?
167262          JAF   BRSR1             % NO - ,CONTINUE
167263          STZ   SPAFL ,B          % RESET SPARE TRACK ALLOCATION FLAG
167264          LDA I (BTSTA
167265  BLCOG,  INIOX LCO               % SET CWR15=0
167266  BRSR1,  INIOX RSR               % READ STATUS
167267          BSKP  ZRO 20 DA         % CONTROLLER ACTIV?
167270  L39,    JMP I (EXBUS            % YES, BUSY EXIT
167271          LDA   BUSFL,B           % PREVIOUS TRANSFER TO BE CHECKED?
167272          JAF   BRSR2             % YES. SKIP UNIT SELECT
167273          LDA   TRG ,B
167274          AND   (0700
167275          SHA   ZIN 1
167276          BSET  ONE 30 DA
167277          JPL   BLCO1             % SELECT UNIT  (TEST MODE)
167300          AAA   010
167301          JPL   BLCO1             % CLEAR DEVICE
167302  BRSR2,  INIOX RSR               % READ STATUS
167303          STA I (SSTAT            % SAVE IT
167304          BSKP  ONE 160 DA        % ON CYLINDER?
167305          JMP I (ERCYL            % NO. ERROR
167306          LDA   BUSFL,B
167307          JAZ   L18               % ZERO: NO PREVIOUS TRANSFER. JUMP TO BCONT
167310          JMP   I *+1             % JUMP TO SPARE TRACK ALLOCATION HANDLER
167311          SPHAN
167312  L33,    LDA   SMARG,B           % ERROR RECOVERY CYCLE ?
167313          JAF   SECPR             % NONZERO: YES. JUMP TO SECTOR PROCESSING
167314          LDA I (SSTAT            % TEST STATUS
167315          AND   (037760
167316          JAF   BREX              % NONZERO: ERRORS IN THE PREVIOUS TRANSFER
167317          JPL I (BRSR6            % READ AND CHECK THE CORE ADDR REG
167320          JMP I (ECADR
167321  L18,    JMP I (BCONT            % NO ERRORS, CONTINUE
167322
167322  BREX=*
167322
167322  "8BDIS+8BDIM
167322
167322          LDA   SVLCO,B           % PREVIOUS CONTROL WORD
167323          SHA   SHR 13
```

```
167324          AAA    -1
167325          JAZ    BWTRY            % ZERO: PREVIOUS DEVICE OP. WAS WRITE
167326
167326  "8BDIS+8BDIM+8ZBDI
167326
167326          LDA    TYPEC ,B         % IS ERROR CORRECTION ALLOWED ?
167327          BSKP   ZRO 020 DA
167330          JMP I (BRTRY            % NO
167331
167331  % PREPARE ERROR RECOVERY BY INITIATING SECTOR-BY-SECTOR PROCESSING (NOT WRITE)
167331
167331          SAA    -1
167332          STA    SMARG,B          % SET SECTOR RECOVERY FLAG
167333          LDD    SVLWC-1 ,B       % GET LEAST WORD COUNT TO D-REG
167334          LDA    SVLWK ,B         % MOST SIGN WORD COUNT
167335          LDX I (BSECW
167336          RDIV   SX               % CONVERT TO NUMBER OF SECTORS
167337          LDX    XRG ,B
167340          COPY   SA DA CM2
167341          COPY   SA DA AD1
167342          STA    SLONG,B          % STORE SECTOR CONTROL NUMBER
167343          LDA I (BSECW
167344          STA    SVLWC,B          % SET WORD COUNT TO ONE SECTOR
167345          STZ    SVLWK ,B
167346          JMP I (BRTRY
167347
167347  "8BDIM+8BDIS
167347
167347  % WRITE-RETRY AFTER ERROR
167347
167347  BWTRY, MIN     ERRC1,B
167350          JMP    L8
167351          JMP I (ERR              % NO MORE RETRIES ALLOWED
167352  L8,     MIN    SWTRY,B          % COUNT ONE RETRY
167353          0
167354          JMP I (BOM
167355
167355  "8BDIM+8BDIS+8ZBDI
167355
167355  )FILL
167400
167400  BLCO1, INIOX LCO                % LOAD CONTROL WORD
167401          EXIT
```

```
167402
167402  % ERROR RECOVERY (SECTOR-BY-SECTOR PROCESSING)
167402  %
167402  SECPR,  LDA I (SSTAT            % GET STATUS
167403          AND   (037760
167404          JAF   ERRAN             % NONZERO: ERROR DURING PREVIOUS TRANSFER
167405  L4,     JPL I (BRSR6            % READ AND CHECK THE CORE ADDR REG
167406          JMP I (ECADR
167407          LDA   SLONG,B           % TEST SECTOR RETRY NUMBER
167410          JAP   L18               % POS: SECTOR RETRY COMPLETED. JUMP TO BCONT
167411          SAA   037
167412          AND   SVLBA ,B          % MASK OUT SECTOR BITS
167413          SUB I (BSECT            % SECTORS/TRACK
167414          AAA   1                 % FIRST=0 NOT 1!!
167415          JAF   L10               % JUMP IF NOT LAST SECTOR ON TRACK
167416          LDA   SVLBA,B
167417          AND   (17400            % MASK FOR HEAD BITS
167420          ADD   (400              % INCR. HEAD ADDR.
167421          STA   SVLBA,B
167422          JMP   L11
167423  L10,    MIN   SVLBA,B           % INCREMENT SECTOR ADDRESS
167424  L11,    MIN   SLONG,B           % INCREMENT SECTOR CONTROL NUMBER
167425          0
167426          LDA   CMAD2,B
167427          ADD I (BSECW            % ADD ONE SECTOR
167430          STA   SVLCA,B           % NEW CORE ADDR.     (16 LOWER)
167431          STA   CMAD2,B
167432          LDA   CMAD1,B
167433          COPY  SA DA ADC
167434          STA   CMAD1,B           % NEW CORE ADDR.    (BANK BITS)
167435          JPL I (BQERR            % SUCCESSFULL SECTOR RETRY. RESET ERROR COUNTERS
167436          STZ   MARGC,B           % RESET MARGINAL CYCLE FLAG
167437          LDA   SVLCO,B
167440          BSET  ZRO 120 DA
167441          STA   SVLCO,B           % RESET COMMAND WORD
167442          AND   (017600
167443          STA I (BTSTA
167444          JMP I (BRTRY
```

```
167445
167445  % PREPARE FOR ERROR CORRECTION (CHECK IF CONDITIONS ARE FAVORABLE)
167445  %
167445  ERRAN=*
167445
167445  "8BDIS+8BDIM
167445
167445          LDA     SVLCO,B
167446          SHA     SHR 13
167447          AAA     -3
167450          JAZ     RETRY           % JUMP IF COMPARE COMMAND
167451          BSKP    ZRO 0140
167452          JMP     L24             % N100
167453          LDA     CMAD1,B         % TEST FOR CORE ADDR. > 256K
167454          SHA     ZIN SHR 2
167455          JAF     RETRY           % CORE ADDR. TOO BIG FOR ECC
167456
167456  "8BDIS+8BDIM+8ZBDI
167456
167456  L24,    LDA I   (SSTAT          % GET STATUS
167457          BSKP    ONE 110 DA      % SKIP IF DATA ERROR
167460          JMP     RETRY           % IF NOT --RETRY.
167461          AND     (036740         % MASK FOR ALL ERRORS (OTHERS)
167462          JAF     RETRY           % JUMP TO RETRY IF ANY ERRORS
167463  BRSC5,  INIOX   RSC             % READ SEEK CONDITION
167464          BSKP    ZRO 170 DA      % SKIP IF NOT ADDRESS FIELD
167465          JMP     RETRY           % ADDRESS FIELD,-GO TO RETRY
167466          COPY    SA DT           % SAVE SEEK CONDITION
167467          LDA     ECCFL,B         % IS ECC OP. FLAG SET?
167470          JAZ     RECOP           % JUMP IF FLAG NOT SET
167471          BSKP    ZRO 150 DT      % SKIP IF MAX ECC COUNT
167472          JMP     ECCOP           % CORRECTABLE !
167473          STZ     ECCFL,B         % PREPARE FOR RETRY
167474          LDA I   (BTSTA          %
167475          BSET    ONE 170 DA
167476          JPL     BLC01           % SET CWR15=1.
167477  BRSC3,  INIOX   RSC             % READ ECC COUNT
167500          SUB     (20071          % COUNT SHOULD BE =8249
167501          JAF     ERREC
167502  RETRY,  JPL I   (RESEC          % RESET ECC
167503          JMP I   (BRTRY          % MARGINAL RECOVERY CYCLE EXIT
167504  RECOP,  SAA     -1
167505          STA     ECCFL,B
167506          LDA I   (BTSTA
167507          BSET    ZRO 0140 DA     % REMOVE BIT 12 IF PARITY CHECK
167510          ADD     (40005          % SET CWR14=0,ACTIVATE,ENBL.INTRPT.
167511          JPL     BLC01           % START ECC-OP.
167512          JMP I   (EXBUS
167513
167513  )FILL
```

```
167534
167534  % ERROR CORRECTION
167534  %
167534  ECCOP,  STZ    ECCFL,B          % ECCFL:=0
167535          LDA I  (BTSTA
167536          BSET   ONE 170 DA
167537          JPL    BLCO1            % SET CWR15=1.
167540  BRSR4,  INIOX  RSR              % READ ECC PATTERN
167541          SAT    -13
167542          STT    CDISP ,B
167543          SAT    0
167544          JMP    L2               % START PATTERN MIRRORING
167545  L1,     SHA    SHR 1
167546          SHT    ZIN 1
167547  L2,     BSKP   ZRO 0 DA
167550          BSET   ONE 0 DT
167551          MIN    CDISP ,B
167552          JMP    L1
167553          COPY   ST DA
167554          STA    CPAT1,B          % STORE PATTERN TEMPORARILY
167555          JAZ    ERREC
167556  BRSC4,  INIOX  RSC              % READ ECC COUNT ( 2-020001 FOR BIT IN SECTOR)
167557          JAN    ERREC
167560          STA    CDISP ,B         % SAVE ECC COUNT
167561          AAA    -2
167562          JAN    ERREC            % NEG: COUNT < 2
167563          SUB    K8191
167564          AAA    -071
167565          JAP    ERREC            % POS: COUNT > 8249
167566          AAA    056
167567          JAP    AYPAS            % POS: COUNT > 8203.  DATA ALREADY OK
167570          LDA I  (BTSTA
167571          JPL    BLCO1            % SET CWR15=0.
167572          LDA    CDISP ,B         % ECC COUNT
167573          AAA    -014
167574          RCLR   DL               % L:=0
167575          JAP    L6
167576          RINC   DL               % L:=1
167577          AAA    012              % WORDNO MUST NOT BE NEGATIVE
167600  L6,     SAD    SHR 4
167601          STA    CDISP ,B         % WORDNO IN SECTOR
167602          SAA    0
167603          SAD    4
167604          COPY   SA DD
167605          SAA    0105
167606          SKP    DL EQL 0
167607          AAA    012              % BECAUSE OF WORDNO COMPENSATION ABOVE
167610          RSUB   SD DA
167611          ADD    SADIN            % SAD 0
167612          COPY   SA DX
167613          LDA    CPAT1 ,B         % ECC PATTERN
167614          RCLR   DD
167615          EXR    SX               % SHIFT PATTERN
167616          STA    CPAT1 ,B         % SAVE CORRECTION PATTERNS
167617          COPY   SD DA
167620          STA    CPAT2 ,B
167621
167621  .BBDIM+BBDIS
167621
167621          LDA I  (BTSTA
```

```
167622          BSKP   ZRO 0140 DA      % TEST IF PARITY COMMAND
167623          JMP    AYPAS            % BYPASS CORRECTION ON PARITY COMMAND
167624
167624  '8BDIM+8BDIS+8ZBDI
167624
167624          LDD    CMAD1-1 ,B       % CMAD1 TO D
167625          LDA    CDISP ,B         % WORDNO IN SECTOR
167626          ADD    CMAD2 ,B
167627          SWAP   SA DD
167630          COPY   SA ADC DA
167631          STD    ADSAV            % SAVE CORRECTION ADDR
167632          LDX    CPAT1 ,B         % PATTERN 1
167633          JXZ    L5
167634
167634  '8BDIM+8BDIS
167634
167634          JPL I (PLDWO            % GET DATA
167635
167635  "8BDIM+8BDIS+8ZBDI
167635
167635          REXO   SX DT            % CORRECT DATA
167636
167636  '8BDIM+8BDIS
167636
167636          LDD    ADSAV            % 24-BIT ADRESS
167637          JPL I (PSTWO            % STORE CORRECTED DATA
167640
167640  "8BDIM+8BDIS+8ZBDI
167640
167640  L5.     LDD    ADSAV            % 24-BIT ADRESS
167641          RINC   DD
167642          COPY   ADC SA DA
167643          STD    ADSAV            % SAVE NEXT 24 BIT ADRESS
167644          LDX    CDISP ,B         % WORDNO IN SECTOR
167645          AAX    1
167646          BSKP   ZRO 0110 DX
167647          JMP    AYPAS            % WORDNO IS 01000.  DONT CORRECT
167650          LDX    CPAT2 ,B         % PATTERN 2
167651          JXZ    AYPAS
167652
167652  '8BDIM+8BDIS
167652
167652          JPL I (PLDWO            % NEXT DATA WORD
167653
167653  "8BDIM+8BDIS+8ZBDI
167653
167653          REXO   SX DT            % CORRECT DATA
167654
167654  "8BDIM+8BDIS
167654
167654          LDD    ADSAV            % 24-BIT ADRESS
167655          JPL I (PSTWO            % STORE CORRECTED DATA
167656
167656  '8BDIM+8BDIS+8ZBDI
167656
167656  AYPAS.  MIN    CORCU ,B         % INCREASE CORRECTION COUNTER
167657          U
167660          LDA I (SSTAT            % REMOVE DATA ERROR BIT FROM STATUS
167661          AND    (176757
167662          STA I (SSTAT
```

```
167663          LDD    ARG ,B          % RESET AD
167664          LDX    XRG ,B
167665          JPL I (RESEC           % RESET ECC
167666          JMP I (L4
167667
167667  ERREC,  JMP I (ERREB           % JUMP-HELP
167670
167670  K8191,  017777
167671  SADIN,  SAD    0
167672  ADSAV,  0; 0
167674
167674  )FILL
```

```
167704
167704  % TRY AGAIN AFTER ERROR (NOT WRITE)
167704  %
167704  BRTRY,  LDA    MARGC,B          % MARC. REC. CYCLE FLAG ALREADY SET?
167705          JAF    BMARG            % NONZERO: YES
167706          STZ    ECCFL,B          % RESET ECC PROCESSING FLAG
167707          MIN    ERRC1,B          % ORDINARY RETRIES DONE?
167710          JMP    L12              % NO. TRY ONCE MORE
167711          LDA    TYPEC ,B         % MARG REC ALLOWED ?
167712          BSKP   ZRO 030 DA
167713          JMP I  (ERR             % NO
167714          ADD    (02000           % INCREMENT MARG REC COUNTER (BITS 15-10)
167715          STA    TYPEC ,B
167716          SAA    -1
167717          STA    MARGC,B          % SET MARG REC. CYCLE FLAG
167720          LDA    SVLCO,B
167721          BSET   ONE 120 DA       % SET BIT 10 IN CONTROL WORD(MARG. REC.)
167722          STA    SVLCO ,B
167723          AND    (017600
167724          STA I  (BTSTA
167725  BMARG,  MIN    ERRC2,B          % ALL MARG. REC.'S DONE?
167726          JMP    BOM
167727          JMP I  (ERR             % 3 MARG. REC. CYCLES DONE. ERROR
167730  L12,    MIN    SRTRY ,B
167731          0
167732  %
167732  % TRANSFER ROUTINE
167732  %
167732  BOM,    LDA I  (BTSTA
167733          BSET   ONE  40 DA       % CLEAR DEVICE   ***NEW,IMPROVED!***
167734          JPL    BLCOF            % CWR 15:=0
167735  BRSR8,  INIOX  RSR              % CLEAR UPPER/LOWER CORE ADDR. FLIP-FLOP
167736          LDA    SVLBA,B          % HEAD AND SECTOR
167737
167737  "8BDIS+8BDIM+8ZBDI
167737
167737  BLBA1,  INIOX  LBA              % LOAD HEAD AND SECTOR
167740          LDA I  (BTSTA
167741          BSET   ONE 170 DA
167742          JPL    BLCOF            % CWR 15:=1
167743          LDA    SVLBB,B
167744
167744  "8BDIS+8BDIM
167744
167744          SAA    077
167745          AND    TRG ,B
167746          AAA    -043
167747          JAZ    L13
167750          AAA    043-044
167751          JAZ    L13
167752          SAA    0
167753          BSET   ONE 0170 DA
167754  L13,    BSET   BCM 0170 DA
167755          ADD    SVLBB ,B
167756
167756  "8BDIS+8BDIM+8ZBDI
167756
167756  BLBA3,  INIOX  LBA              % LOAD CYLINDER ADDRESS
167757          JPL I  (RESEC           % RESET ECC.  SET CWR15:=0
167760          LDA    CMAD1,B
```

```
167761  BLCA1,  INIOX LCA               % LOAD BANK NO
167762          LDA   CMAD2,B
167763  BLCA3,  INIOX LCA               % LOAD MEMORY ADDR WITHIN BANK
167764          LDA   SVLWK ,B          % MOST SIGN WORD COUNT
167765  BLWC2,  INIOX LWC
167766          LDA   SVLWC,B
167767  BLWC1,  INIOX LWC               % LOAD WORD COUNT
167770          LDA   SVLCO,B
167771          JPL   BLCOF             % START TRANSFER
167772  BRSR3,  INIOX RSR               % READ STATUS
167773          STA I (SSTAT
167774          BSKP  ONE 20 DA         % CONTROLLER ACTIVE ?
167775          JMP I (ERACT            % NO
167776          LDA   SVLWC ,B          % WORD COUNT
167777          ADD   CMAD2 ,B          % LEAST SIGN. PART OF MEMORY ADDR
170000          STA   SCADR ,B          % EXP. LEAST SIGN. PART OF THE CORE ADDR REG
170001  EXBUS,  SAA   -1
170002          STA   BUSFL,B           % SET BUSY FLAG
170003          LDA I (9LREG
170004          COPY  SA DL
170005          LDF   TRG ,B
170006          EXIT  AD1               % BUSY EXIT!!!
170007
170007  )FILL
```

```
170017
170017  % START A NEW TRANSFER WHEN X IS NONZERO
170017  %
170017  BCONT, SKP    IF DX UEQ 0
170020         JMP I  (FINEX          % ALL BLOCKS TRANSFERED
170021         STZ    SMARG,B
170022  BRSC6, INIOX  RSC             % READ SEEK COND.
170023         COPY   SA DT
170024         LDA I  (BTSTA          %
170025         BSET   ONE 170 DA
170026         JPL    BLCOF           % SET CWR 15 =1.
170027         LDA    TYPEC ,B
170030         COPY   SA DL
170031         BSET   ONE 0110 DL     % BIT FOR 15KHZ INTERFACE
170032  BRSR5, INIOX  RSR             % READ ECC PATTERN
170033         AND    (174000         % MASK FOR BITS 11-15 OF A-REG.
170034         SUB    (134000
170035         JAZ    NEWIE           % ZERO: 15KHZ INTERFACE
170036         BSET   ZRO 0110 DL
170037         BSET   ZRO 0160 DA
170040         JAZ    NEWIE           % ZERO: 10 KHZ INTERFACE
170041         JMP I  (ERTYP          % ERROR.
170042  %
170042  % DECIDE IF HEAD-ADVANCE SHOULD BE USED
170042  %
170042  NEWIE, COPY   SL DA
170043         STA    TYPEC ,B
170044         COPY   AD1 DL          % L:=1
170045
170045  "8BDIS+8BDIM
170045
170045         LDA    TRG ,B
170046         BSKP   ZRO 0160 DA
170047         JMP    L19             % PHOENIX
170050         SHA    075
170051         AND    (030            % UNIT NUMBER IN BITS 4-3
170052         ADD    (BSKP  ZRO 040 DA
170053         STA    *+2
170054         LDA    TYPEC ,B        % IS HEAD ADVANCE ALLOWED ?
170055         0                      % BSKP  ZRO 040 DA   FOR UNIT 0
170056         JMP    L19             % NO
170057         BSKP   ZRO 0140 DT
170060         BSET   ONE 010 DL      % HEAD ADVANCE. L:=3
170061
170061  "8BDIS+8BDIM+8ZBDI
170061
170061  L19,   SAA    -4
170062         AND    TYPEC ,B        % REMOVE BITS 1-0 FROM TYPEC
170063         RADD   SL DA
170064         STA    TYPEC ,B        % 1 OR 3 IN BITS 1-0
170065  %
170065  % COMPUTE THE BLOCK ADDRESS OF THE LAST BLOCK (SECTOR) IN THE TRANSFER
170065  %
170065         LDA    ARG ,B          % MOST SIGN. OF LOG. DISK ADDR
170066         LDT    TRG ,B
170067         BSKP   ONE 0160 DT     % PHOENIX DISK ?
170070         JMP    XY              % NO
170071         BLDA   170 DD          % SAVE FIXED/REMOV INDICATOR IN K
170072         BSET   ZRO 170 DD
170073         SAT    0
```

```
170074          BSET   BAC 140 DT      % SURFACE ALWAYS 20+N FOR FIXED DISK
170075          STT    SAVT
170076          JMP    XY
170077
170077  SAVT,   0
170100
170100  )FILL
170107  K7000, 07000
170110
170110  BLCOF,  INIOX LCO              % LOAD CONTROL WORD
170111          EXIT
170112
170112  XY,     AAX    -1
170113          RADD   SX DD
170114          COPY   SA ADC DA       % ADDR OF LAST SECTOR NOW IN AD
170115          AAX    1
170116          LDT I  (BSECY          % SECTORS/CYL
170117          RDIV   ST
170120          LDT I  (BMXCY
170121          SKP    DT MGRE SA
170122  L40,    JMP I  (ILADR          % CYLINDER > MAX CYLINDER
170123          STA    SVLBB,B         % SAVE CYLINDER ARGUMENT
170124          COPY   SD DA AD1
170125          STA    HESEC           % 0 < HESEC <= BSECY
170126          SKP    DX GRE SA       % CHECK IF TRANSFER IS GREATER THAN 63.5K
170127          COPY   SX DA
170130          LDT I  (BSECW          % A IS NO. OF SECTORS IN TRANSFER
170131          RMPY   ST DA
170132          JAZ    L20             % ZERO: LESS THAN 64K
170133          SWAP   SA DD
170134          RADD   ST DA
170135          RADD   ADC CM1 DD
170136          SWAP   SA DD
170137          RDIV   ST              % AFTER THIS, A HOLDS NUMBER OF SURPLUS SECTORS
170140          LDT    TYPEC ,B
170141          BSKP   ZRO 0110 DT
170142          SAA    0               % 15 KHZ INTERFACE. ALLOW MORE THAN 63.5K
170143  L20,    COPY   SA DL           % SAVE IT IN L FOR A LITTLE WHILE
170144          LDA    HESEC
170145          COPY   SA DD
170146  %
170146  % COMPUTE SURFACE AND SECTOR NUMBER
170146  %
170146          LDA    TYPEC,B
170147          BSKP   ONE 10 DA       % SKIP IF HEAD ADVANCE
170150          JMP    L14
170151          COPY   SX DA
170152          SKP    IF DA LST SD    % SKIP IF WHOLE TRANSFER WITHIN CYLINDER
170153          COPY   SD DA
170154          RSUB   SA DD
170155          RADD   SL DD           % SURPLUS SECTORS
170156          JMP    L23
170157  L14,    RDCR   DD
170160  L23,    SAA    0
170161          LDT I  (BSECT          % GET SECTORS/TRACK
170162          RDIV   ST              % A:=SURFACE NO.
170163          COPY   SD DT           % D&T=NUMBER OF SECTORS IN LAST CYLINDER
170164          SHA    ZIN  10         % SURFACE IN LEFT BYTE
170165          RADD   SA DD           % MERGE SURFACE AND SECTOR
170166          AAT    1               % NO. OF SECTORS IN LAST CYLINDER
```

```
170167          LDA   TRG ,B
170170          BSKP  ONE 160 DA        % IF CARTRIDGE DISK
170171          JMP   NOTPH
170172          AND   K7000             % MASK FOR SURFACE (LA BIT 16-18)
170173          SHA   ZIN SHR 1         % MOVE TO BIT 8-10
170174          ADD   SAVT              % ADD SECTOR AND FIXED/REMOVABLE SURFACE FLAG
170175          ADD   HESEC
170176          AAA   -1
170177          COPY  SA DD
170200          AND   (03400
170201          JAZ   NOTPH             % ZERO: SURFACE 0 OR 020
170202          BSKP  ONE 0140 DD
170203          JMP   L40               % REMOVABLE, AND SURFACE # 0 !!
170204  NOTPH,  LDA   TYPEC,B
170205          BSKP  ONE 10 DA         % SKIP IF HEAD ADVANCE
170206          JMP   L15
170207          COPY  SD DA
170210          JMP   L3
170211  L15,    SKP   IF DX GRE ST
170212          COPY  SX DT             % T: NO. OF SECTORS TO TRANSFER
170213          COPY  SD DA AD1
170214          RSUB  ST DA             % A: HARDWARE BLOCK ADR.
170215  L3,     STA   SVLBA,B           % SAVE HEAD AND SECTOR
170216  %
170216  % DECREMENT X
170216  %
170216          LDA   TYPEC,B
170217          BSKP  ONE 10 DA
170220          JMP   L7                % JUMP IF NOT HEAD ADVANCE
170221          LDT   HESEC             % LOAD HEAD + SECTOR
170222          SKP   IF DX GRE ST
170223          COPY  SX DT
170224          RSUB  SL DT             % SURPLUS SECTORS
170225  L7,     RSUB  ST DX             % X: NO. OF BLOCKS LEFT TO THE NEXT TRANSFER
170226          STX   XRG ,B
170227  %
170227  % X IS NOW NO. OF BLOCKS TO BE TRANSFERRED NEXT TIME.
170227  % T IS NOW NO. OF BLOCKS TO TRANSFER THIS TIME.
170227  % COMPUTE SVLCA, CMAD1, CMAD2, SVLWC.
170227  %
170227          LDA I (BSECW
170230          RMPY  SX DA             % AD: X*BSECW=NO. OF WORDS LEFT
170231          SWAP  SA DD
170232          ADD   MEMA2 ,B
170233          COPY  SD ADC DD
170234          STA   SVLCA ,B
170235          SWAP  SA DD
170236          ADD   MEMA1 ,B
170237          STD   CMAD1 ,B
170240          LDA I (BSECW
170241          RMPY  ST DA             % AD: T*BSECW=NO. OF WORDS TO TRANSFER
170242          STA   SVLWK ,B          % SAVE MOST SIGN WORD COUNT
170243          COPY  SD DA
170244          STA   SVLWC,B           % SAVE WORD COUNT
170245  %
170245  % PREPARE BTSTA AND THE CONTROL WORD (SVLCO)
170245  %
170245  "BBDI5+8BDIM
170245
170245          SAA   3
```

```
170246          AND    CMAD1 ,B
170247          SHA    5                % CORE ADDR BITS 17-16
170250          COPY   SA DT
170251
170251  "8BDIS+8BDIM+8ZBDI
170251
170251          LDA    TRG ,B
170252          AND    K0700
170253          SHA    ZIN 1            % UNIT NO
170254
170254  "8BDIS+8BDIM
170254
170254          RADD   SA DT
170255          SAA    077
170256          AND    TRG ,B
170257          AAA    -042
170260          JAZ    L54              % FUNCTION 042
170261          AAA    042-043
170262          JAZ    L56              % FUNCTION 043
170263          AAA    043-044
170264          JAZ    L55              % FUNCTION 044
170265          AAA    044
170266          JMP    L56
170267  L54,    AAA    1
170270  L55,    AAA    1
170271  L56,    SHA    013              % DEVICE OPERATION
170272          RADD   ST DA
170273
170273  "8BDIS+8BDIM+8ZBDI
170273
170273          STA I  (BTSTA
170274          AAA    5                % ACTIVATE & ENABLE INTERRUPT
170275          STA    SVLC0,B          % SAVE CONTROL WORD
170276
170276  "8BDIS+8BDIM
170276
170276          SAA    077              % CHECK FOR FUNCTION 042 (TEST IF SPARE-SECTOR FORMAT)
170277          AND    TRG ,B
170300          AAA    -042
170301          JAF    L42
170302          JMP I  (L60             % YES, FUNCTION 042
170303
170303  % THE CODE BELOW IS FOR FUNCTION 037, WHICH IS A SUBFUNCTION OF FUNCTION 042.
170303  % IT WILL START A PARITY CHECK ON THE SPARE TRACK AREA.
170303
170303  L65,    LDA I  (BMXCY+1         % FIRST CYL IN SPARE TRACK AREA
170304          STA    SVL8B ,B
170305          JAF    L42
170306          JMP I  (L64             % NO SPARE TRACKS
170307
170307  "8BDIS+8BDIM+8ZBDI
170307
170307  L42,    JMP I  (BOM             % JUMP TO TRANSFER ROUTINE
170310
170310  )FILL
170323  K0700, 0700
170324  HESEC, 0
```

```
170325
170325  % THIS IS THE FINISHED-AND-OK EXIT
170325  %
170325  FINEX, LDX    SSTAT
170326         JPL    BQERR            % RESET ERROR COUNTERS
170327         STZ    ECCFL,B
170330         STZ    SMARG,B
170331         STZ    MARGC,B
170332         STZ    BUSFL,B
170333         STZ    SPAFL ,B
170334
170334  "8BDIS+8BDIM
170334
170334         JPL I (L52              % AUTOMATIC RELEASE
170335
170335  "8BDIS+8BDIM+8ZBDI
170335
170335         LDA    9LREG
170336         COPY   SA AD1 DL
170337         LDF    TRG ,B
170340         EXIT   AD1              % FINISHED EXIT
170341  %
170341  % RESET-ECC SUBROUTINE
170341  %
170341  RESEC, LDA    BTSTA
170342         BSET   ONE 170 DA
170343  BLCOH, INIOX LCO               % SET CWR15 = 1
170344         SAA    1
170345         ADD I (BMXCY+2
170346  BLWC3, INIOX LWC               % RESET ECC
170347         LDA    BTSTA
170350  BLCO8, INIOX LCO               % SET CWR15=0.
170351         EXIT
170352  %
170352  % READ-AND-CHECK THE CORE ADDR REG
170352  %
170352  BRSR6, INIOX RSR               % CLEAR UPPER/LOWER CORE ADDR. FLIP-FLOP
170353
170353  "8BDIS+8BDIM
170353
170353         SAA    077
170354         AND    TRG ,B
170355         AAA    -2
170356         JAZ    L38              % ZERO: PARITY CHECK
170357
170357  "8BDIS+8BDIM+8ZBDI
170357
170357  BRCA,  INIOX RCA               % READ CORE ADDR. REG
170360         STA    DRIAR
170361         SUB    SCADR,B
170362         JAF    L37              % NONZERO: CORE ADDR. REG NOT AS EXPECTED
170363         LDA    SVLWC ,B         % WORD COUNT
170364         ADD    CMAD2 ,B         % LEAST MEM ADDR
170365         LDA    CMAD1 ,B         % BANK NO
170366         COPY   SA ADC DA        % EXP. BANK NO
170367         ADD    SVLWK ,B         % MOST SIGN WORD COUNT
170370         STA    SAVBN
170371  BRCA2, INIOX RCA               % READ BANK NO
170372         AND    K0377
170373         SUB    SAVBN
```

```
170374          STA    SAVBN            % SAVE RESULT TEMP.
170375  BRSC1,  INIOX  RSC
170376          BSKP   ONE 0140 DA
170377  L38,    EXIT   AD1              % N10 INTERFACE, OR OK
170400          LDA    SAVBN
170401          JAZ    L38
170402  L37,    EXIT                    % BANK NO, OR ADDR, NOT AS EXPECTED
170403
170403  SAVBN,  0
170404  K0377,  0377
170405
170405  )FILL
170407
170407  % RESET ERROR COUNTERS
170407
170407  BQERR,  SAA    -4
170410          STA    ERRC1 ,B
170411          SAA    -034
170412          STA    ERRC2 ,B
170413          EXIT
```

```
170414
170414  % BELOW FOLLOW THE DIFFERENT ERROR EXITS
170414  %
170414  ERACT,  SAT     0100            % NOT ACTIVE AFTER ACTIVATE
170415          JMP     L9              % SET BIT 017 IN T
170416
170416  ILCOD,  SAT     040             % ILLEGAL DEVICE OPERATION
170417          JMP     L9              % SET BIT 016 IN T
170420
170420  ERTYP,  SAT     020             % CONTROLLER DEFINITION PROBLEM
170421          JMP     L9              % SET BIT 015 IN T
170422
170422  ERREB,  SAT     010             % ECC PROBLEM
170423          JMP     L9              % SET BIT 014 IN T
170424
170424  ILADR,  SAT     4               % LOGICAL DISK ADDR IS TOO BIG
170425          JMP     L9              % SET BIT 013 IN T
170426
170426  ECADR,  SAT     2               % CORE ADDR REG ERROR. SET BIT 012 IN T
170427
170427  L9,     SHT     011             % SHIFT ERROR BIT TO CORRECT PLACE
170430          LDA     TRG ,B
170431          AND     BMASK
170432          RADD    SA DT
170433          SAA     0
170434          JPL     BLC08           % CWR15:=0
170435  BRSC7,  INIOX   RSC
170436          STA     RSCON
170437          JPL     BQERR           % RESET ERROR COUNTERS
170440          JMP     ERREK
170441
170441  ERCYL,  SAX     0               % NOT ON-CYLINDER
170442          JMP     L36             % CONTINUE WITH X=0
170443
170443  ERR,    JPL     BQERR           % ALL ERRC1 (WRITE) OR ERRC2 (NON-WRITE) TRANSFERS ARE DONE
170444          SAX     -1              % SET X TO NONZERO
170445  L36,    LDA     TRG ,B
170446          AND     BMASK
170447          COPY    SA DT
170450  BRSC2,  INIOX   RSC
170451          STA     RSCON
170452          BSKP    ZRO 130 DA
170453          BSET    ONE 110 DT      % SEEK ERROR
170454          BSKP    ONE 130 DA
170455          JXZ     ERREK           % JUMP TO ERREK IF X=0 AND NOT SEEK ERROR
170456          LDA     TRG ,B
170457          AND     K0700
170460          SHA     ZIN 1
170461          AAA     20
170462  BLC03,  INIOX   LCO             % DEVICE CLEAR
170463          ADD     (033775         % RETURN-TO-ZERO SEEK (TEST MODE)
170464          LDX     RSCON
170465          BSKP    ONE 0140 DX
170466          BSET    ZRO 030 DA      % N-10.  REMOVE TEST MODE
170467  BLC04,  INIOX   LCO
170470  ERREK,  STZ     BUSFL,B
170471          STZ     ECCFL,B
170472          STZ     SMARG,B
170473          STZ     MARGC,B
170474          STZ     SPAFL ,B
```

```
170475
170475  "8BDIS+8BDIM
170475
170475          JPL I (L52              % AUTOMATIC RELEASE
170476
170476  "8BDIS+8BDIM+8ZBDI
170476
170476          LDX    SSTAT
170477
170477  "8BDIS+8BDIM
170477
170477          STX    QQQST            % DISK STATUS
170500          STT    TREGI            % ERROR INFORMATION
170501          LDA    SVLCO ,B
170502          STA    CONTR            % CONTROL WORD
170503          LDA    HDEV ,B
170504          STA    DEVNO            % DEVICE NUMBER
170505          LDA    CMAD1 ,B
170506          STA    BANKN            % BANK NUMBER
170507          LDA    CMAD2 ,B
170510          STA    ADRES            % MEMORY ADDRESS
170511          LDA    SVLBB ,B
170512          STA    QQQCY            % DISK CYLINDER
170513          LDA    SVLBA ,B
170514          STA    SRFSC            % DISK SURFACE AND SECTOR
170515          LDA    SVLWC ,B
170516          STA    WORDC            % WORD COUNT
170517
170517  "8BDIS+8BDIM+8ZBDI
170517
170517          LDA    ARG ,B
170520          JMP I  9LREG            % EXIT
170521
170621  )FILL
170523
170523  SSTAT, 0
170524  BTSTA, 0
170525  BMASK, 0777
170526  9LREG, 0
170527
170527  % BELOW FOLLOW 11 WORDS THAT CONTAIN ERROR INFORMATION ABOUT THE LAST ERROR.
170527  % THEY ARE STORED THERE WHEN THE DRIVER ERROR EXITS. PLEASE NOTE THAT THE
170527  % LOWER PART OF THE CORE ADDRESS REGISTER IS STORED IN DRIAR EVERY TIME THAT
170527  % REGISTER IS CHECKED, AND THIS IMPLIES THAT THE STATUS REGISTER MUST BE OK.
170527
170527  DRIAR, 0      % LOWER 16 BITS OF CORE ADDR REG AFTER TRANSFER (STATUS IS OK)
170530
170530  "8BDIM+8BDIS
170530
170530  QQQST, 0      % STATUS REGISTER
170531
170531   8BDIM+8BDIS+8ZBDI
170531
170531  RSCON, 0      % SEEK CONDITION
170532
170532  "8BDIM+8BDIS
170532
170532  TREGI, 0      % BITS 017-011 ARE ERROR INDICATORS, 010-6 UNIT, 5-0 DEV. OP.
170533  CONTR, 0      % CONTROL WORD
170534  DEVNO, 0      % DEVICE NUMBER (01540 OR 01550)
```

```
170535  BANKN, 0       % 8 MOST SIGNIFICANT BITS OF THE MEMORY ADDRESS
170536  ADRES, 0       % 16 LEAST SIGNIFICANT BITS OF THE MEMORY ADDRESS
170537  QQQCY, 0       % CYLINDER              PART OF THE DISK ADDRESS
170540  SRFSC, 0       % SURFACE AND SECTOR PART OF THE DISK ADDRESS
170541  WORDC, 0       % WORD COUNT
170542
170542  "8BDIS+8BDIM+8ZBDI
170542
170542  % SPARE TRACK ALLOCATION HANDLER.
170542  % ACTIVATED WHEN A TRANSFER GIVES ADDRESS MISMATCH STATUS ERROR.
170542  % CHECKS BIT 4 IN TYPEC ,B TO SEE IF SPARE TRACK ALLOCATION IS ALLOWED.
170542  % THE SPARE TRACK ALLOCATION PARAMETERS (I.E. THE NEW CYLINDER AND SURFACE)
170542  % ARE READ FROM THE BAD TRACK, BY READING WITH THE SAME CYLINDER AND SURFACE
170542  % AS IN THE ORIGINAL ACCESS, BUT WITH SECTOR 0100, 0101, AND SO ON, UNTIL
170542  % TWO WORDS ARE READ CORRECTLY. THOSE TWO WORDS (CYLINDER AND SURFACE) ARE THEN
170542  % USED, TOGETHER WITH THE ORIGINAL SECTOR INCREMENTED BY 0100, TO ACCESS THE
170542  % SPARE TRACK.
170542
170542  SPHAN=*
170542
170542  "8BDIS+8BDIM
170542
170542          SAA    077
170543          AND    TRG ,B
170544          AAA    -042
170545          JAF    *+2
170546          JMP I (L62              % FUNCTION IS TEST-IF-SPARE-TRACK
170547          AAA    042-037
170550          JAF    *+2
170551          JMP I (L66
170552
170552  "8BDIS+8BDIM+8ZBDI
170552
170552          LDA    SPAFL ,B         % THE SPARE TRACK ALLOCATION FLAG CAN BE 0/1/2
170553          JAF    L27
170554
170554  "8BDIS+8BDIM
170554
170554          LDA    TRG ,B
170555          SHA    075
170556          AND    (030             % UNIT NO NOW IN BITS 4-3
170557          ADD    (BSKP  ONE 040 DA
170560          STA    *+2
170561          LDA    TYPEC ,B         % SPARE TRACK ALLOCATION ALLOWED ?
170562          0                       % BSKP  ONE 040 DA   FOR UNIT 0
170563          JMP I (L33              % NO
170564
170564  "8BDIS+8BDIM+8ZBDI
170564
170564          LDA    SSTAT            % ADDRESS MISMATCH ?
170565          BSKP   ONE 0100 DA
170566          JMP I (L33              % NO
170567  BRSC8,  INIOX RSC               % READ SEEK CONDITION
170570          BSKP   ZRO 0130 DA
170571          JMP    ERR              % SEEK ERROR
170572          SAA    1                % FLAG:=1
170573          STA    SPAFL ,B
170574          LDA    CMAD1 ,B         % SAVE PART OF THE DATA FIELD
170575          STA    OCMD1 ,B
170576          LDA    CMAD2 ,B
```

```
170577          STA   OCMD2 ,B
170600          LDA   SVLBA ,B
170601          STA   OSVBA ,B
170602          LDA   SVLBB ,B
170603          STA   OSVBB ,B
170604          LDA   SVLCO ,B
170605          STA   OSVCO ,B
170606          LDA   SVLWC ,B
170607          STA   OSVWC ,B
170610          LDA   SVLWK ,B
170611          STA   OSVWK ,B
170612          LDA   SVLBA ,B        % SET SECTOR 0100
170613          AND   (177400
170614          BSET  ONE 060 DA
170615          STA   SVLBA ,B
170616          LDA   SVLCO ,B        % PUT READ-OP IN CONTROL WORD
170617          AND   (163777
170620          STA   SVLCO ,B
170621          AND   (177740         % AND IN BTSTA
170622          STA   BTSTA
170623  L26,    SAA   2
170624          STA   SVLWC ,B        % WORD COUNT := 2
170625          STZ   SVLWK ,B
170626          STZ   CMAD1 ,B        % MEMORY ADDR IN DATA FIELD
170627          COPY  SB DA
170630          AAA   NWLBB
170631          STA   CMAD2 ,B        % NWLBB AND NWLBA
170632          JMP I (BOM            % TRANSFER
170633  L27,    AAA   -1
170634          JAF   L31             % NONZERO: SPAFL IS NOT 1
170635          LDA   SSTAT           % STATUS ERROR ?
170636          AND   (017760
170637          JAF   L30             % NONZERO: YES. TRY NEXT SECTOR
170640          JPL I (BRSR6          % NO. CHECK THE CORE ADDR REG
170641          JMP   L30             % CORE ADDR REG ERROR
170642          LDA   NWLBB ,B        % CHECK THE NEW CYLINDER NUMBER
170643          JAN   L30             % NEG: ILLEGAL
170644          SUB I (BMXCY
170645          AAA   -1
170646          JAP   L30             % POS: ILLEGAL
170647          LDA   TRG ,B
170650          BSKP  ZRO 0160 DA
170651          JMP   L28             % PHOENIX
170652          LDA   NWLBA ,B        % CHECK THE NEW SURFACE
170653          JAN   L30             % NEG: ILLEGAL
170654          SHA   070
170655          AAA   1
170656          MPY I (BSECT
170657          SUB I (BSECY
170660          AAA   -1
170661          JAP   L30             % POS: ILLEGAL
170662          JMP   L29             % THE NEW CYL AND SURFACE OK (NOT PHOENIX)
170663
170663  .FILL
170701
170701  L28,    LDA   NWLBA ,B        % CHECK PHOENIX SURFACE
170702          JAN   L30             % NEG: ILLEGAL
170703          SHA   070
170704          JAZ   L29             % ZERO: OK
170705          AAA   -020
```

```
170706          JAN   L30              % NEG: ILLEGAL
170707          AAA   -5
170710          JAP   L30              % POS: ILLEGAL
170711  L29,    SAA   2                % FLAG:=2
170712          STA   SPAFL ,B
170713          LDA   OCMD1 ,B         % RESET DATA FIELD
170714          STA   CMAD1 ,B
170715          LDA   OCMD2 ,B
170716          STA   CMAD2 ,B
170717          LDA   OSVCO ,B
170720          STA   SVLCO ,B
170721          AND   (177740
170722          STA   BTSTA
170723          LDA   OSVWC ,B
170724          STA   SVLWC ,B
170725          LDA   OSVWK ,B
170726          STA   SVLWK ,B
170727          LDA   NWLBB ,B
170730          STA   SVLBB ,B         % NEW CYL
170731          LDA   NWLBA ,B
170732          AND   (177400
170733          STA   NWLBA ,B
170734          SAA   077
170735          AND   OSVBA ,B         % OLD SECTOR (+0100, OF COURSE)
170736          AAA   0100
170737          ADD   NWLBA ,B         % NEW SURFACE
170740          STA   SVLBA ,B
170741          JMP I (BOM             % TRANSFER
170742  L30,    LDA   SVLBA ,B         % TRY NEXT SECTOR, IF POSSIBLE
170743          AAA   1
170744          STA   SVLBA ,B
170745          AND   (0377
170746          SUB I (BSECT
170747          AAA   -0100
170750          JAN   L26              % NEG: TRY NEXT
170751          LDA   OCMD1 ,B         % RESET DATA FIELD
170752          STA   CMAD1 ,B
170753          LDA   OCMD2 ,B
170754          STA   CMAD2 ,B
170755          LDA   OSVBA ,B
170756          STA   SVLBA ,B
170757          LDA   OSVBB ,B
170760          STA   SVLBB ,B
170761          LDA   OSVCO ,B
170762          STA   SVLCO ,B
170763          LDA   OSVWC ,B
170764          STA   SVLWC ,B
170765          LDA   OSVWK ,B
170766          STA   SVLWK ,B
170767          JMP I (ERR
170770  L31,    AAA   -1
170771          SKP   DA EQL 0
170772          JMP I (ERR             % SPAFL IS NOT 2
170773          LDA I (SSTAT
170774          AND   (017760
170775          JAP   L32              % NONZERO: STATUS ERROR
170776          JPL I (BRSR6           % CHECK THE CORE ADDR REG
170777          JMP I (ECADR           % ERROR
171000          LDA   OSVBA ,B         % RESET DISK ADDRS
171001          STA   SVLBA ,B
```

```
171002          LDA   OSVBB ,B
171003          STA   SVLBB ,B
171004          STZ   SPAFL ,B          % FLAG:=0
171005          MIN   SPACO ,B          % INCREMENT COUNTER
171006          0
171007  L32,    JMP I (L33              % BACK TO DRIVER
171010
171010  )FILL
171023
171023  "8BDIS+8BDIM
171023
171023  % PRIORITY-SELECT AND RELEASE IS HANDLED HERE
171023
171023  L49,    LDA   BUSFL ,B          % PRIORITY-SELECT. CHECK IF DRIVER IS BUSY
171024          SKP   DA EQL 0
171025          JMP I (EXBUS
171026          LDA   TRG ,B            % GET UNIT NO
171027          AND   (0700
171030          SHA   1
171031          ADD   (044010
171032          JPL I (BLCO8
171033          ADD   (134010           % GIVING 020 + UNIT NO
171034          JPL I (BLCO8
171035          LDA   TYPEC ,B          % SET RELEASE-NOT-ALLOWED BIT
171036          BSET  ONE 0100 DA
171037          STA   TYPEC ,B
171040          JMP   L51
171041  L50,    LDA   BUSFL ,B          % RELEASE. CHECK IF DRIVER IS BUSY
171042          SKP   DA EQL 0
171043          JMP I (EXBUS
171044          LDA   (044020
171045          JPL I (BLCO8            % RELEASE
171046          LDA   (002010
171047          JPL I (BLCO8            % DESELECT
171050          LDA   TYPEC ,B          % RESET RELEASE-NOT-ALLOWED BIT
171051          BSET  ZRO 0100 DA
171052          STA   TYPEC ,B
171053  L51,    LDX   (040011
171054          STX I (SSTAT
171055          JMP I (FINEX
171056
171056  % AUTOMATIC RELEASE IS HANDLED HERE. IT IS CALLED FROM FINEX AND ERREK
171056
171056  L52,    LDA   TYPEC ,B
171057          0                       % BSKP  ZRO 0100 DA                **************
171060          EXIT                    % AUTOMATIC RELEASE IS NOT ALLOWED
171061          COPY  SL DA
171062          STA   L53               % SAVE L-REG
171063          LDA   (044020
171064          JPL I (BLCO8
171065          LDA   (002010
171066          JPL I (BLCO8
171067          JMP I L53
171070
171070  L53,    0
171071
171071  )FILL
171103
171103  % FUNCTION 042 WILL BE STARTED HERE
171103
```

```
171103  L60,    LDA I (BMXCY+4          % POINTER TO RELATED LAYOUT
171104          JAZ   L63               % ZERO: NO SPARE SECTOR FORMAT
171105          COPY  SA DX
171106          LDA   1 ,X              % SECTORS/TRACK FOR RELATED LAYOUT
171107          SUB I (BSECT            % SECTORS/TRACK FOR ACTUAL LAYOUT
171110          JAN   L61               % NEG: ACTUAL LAYOUT ALREADY NON-S-S
171111          LDT   TRG ,B            % CHANGE LAYOUT TO NON-S-S FORMAT
171112          SHT   072
171113          SAA   7
171114          RAND  ST DA             % UNIT NO
171115          AAA   HTABL
171116          RADD  SB DA
171117          SWAP  SA DX
171120          STA   ,X                % STORE RELATED POINTER IN DATA FIELD
171121          COPY  SA DX
171122          LDF   ,X                % CHANGE LAYOUT
171123          STF I (BSECW
171124          LDF   3 ,X
171125          STF I (BSECW+3
171126          LDD   6 ,X
171127          STD I (BSECW+6
171130  L61,    LDX   XRG ,B
171131          LDA   TYPEC ,B
171132          BSKP  ONE 0110 DA
171133          JMP   L63               % 10MHZ INTERFACE
171134          LDA I (BMXCY+4
171135          JAZ   L63               % ZERO: NO RELATED POINTER
171136          LDA I (BSECT
171137          AAA   -1
171140          STA   SVLBA ,B          % LAST SECTOR
171141          JMP I (BOM
171142
171142  % PARITY CHECK ON LAST SECTOR, CYLINDER 0, SURFACE 0, JUST FINISHED.
171142
171142  L62,    LDA I (SSTAT
171143          BSKP  ONE 0100 DA
171144          JMP   L63               % NOT ADDR MISM.  USE NON-S-S FORMAT
171145          LDT   TRG ,B
171146          SHT   072
171147          SAX   7
171150          RAND  ST DX             % UNIT NO
171151          AAX   HTABL
171152          RADD  SB DX
171153          LDA I (BMXCY+4          % POINTER TO RELATED LAYOUT
171154          STA   ,X                % CHANGE POINTER IN DATA FIELD
171155          COPY  SA DX
171156          LDF   ,X                % CHANGE LAYOUT
171157          STF I (BSECW
171160          LDF   3 ,X
171161          STF I (BSECW+3
171162          LDD   6 ,X
171163          STD I (BSECW+6
171164          LDX   XRG ,B
171165  L63,    LDA   TRG ,B
171166          SHA   075
171167          AND   (030
171170          ADD   (BSET ONE 040 DA
171171          STA   *+2
171172          LDA   TYPEC ,B
171173          O                       % SET BIT IN TYPEC
```

```
171174          STA    TYPEC ,B
171175          LDA    TRG ,B
171176          BSKP   ZRO 0160 DA
171177          JMP    L64              % PHOENIX
171200          SAA    -0100
171201          AND    TRG ,B           % REMOVE FUNCTION 042
171202          AAA    037              % INSERT FUNCTION 037
171203          STA    TRG ,B
171204          STZ    SVLBA ,B         % SURFACE 0, SECTOR 0
171205          LDA    SVLCO ,B
171206          BSET   ZRO 0120 DA
171207          STA    SVLCO ,B
171210          LDA I  (BTSTA
171211          BSET   ZRO 0120 DA
171212          STA I  (BTSTA
171213          JMP I  (L65
171214
171214  % PARITY CHECK ON SPARE TRACK AREA JUST FINISHED
171214
171214  L66,    LDA I  (SSTAT
171215          BSKP   ZRO 0100 DA
171216          JMP    L64              % ADDR MISM
171217          LDA    TRG ,B
171220          SHA    075
171221          AND    (030
171222          ADD    (BSET ZRO 040 DA
171223          STA    *+2
171224          LDA    TYPEC ,B
171225          0                       % RESET BIT IN TYPEC
171226          STA    TYPEC ,B
171227  L64,    SAA    -0100            % RESET FUNCTION 042
171230          AND    TRG ,B
171231          AAA    042
171232          STA    TRG ,B
171233          SAX    0
171234          LDA    TRG ,B
171235          SHA    075
171236          AND    (030
171237          ADD    (BSKP ZRO 040 DA
171240          STA    *+2
171241          LDA    TYPEC ,B
171242          0
171243          AAX    1                % MEANS SPARE TRACKS
171244          LDA I  (BMXCY+4         % POINTER TO RELATED LAYOUT
171245          JAZ    L67
171246          SWAP   SX DA
171247          LDX    1 ,X
171250          SWAP   SX DA
171251          SUB I  (BSECT
171252          JAN    L67
171253          AAX    2                % MEANS SPARE SECTOR FORMAT
171254  L67,    STX I  (SSTAT
171255          JMP I  (FINEX
171256
171256  )FILL
171274  )PCL MORE9
171274  )KILL RCA RSC LBA RSR LCO LWC LCA INIOX MORE9
171274
171274  )BDIS+BBDIM
171274  )KILL BSECW BSECT BSECY BMXCY
```

```
171274
171274  "8BDIS+8BDIM
171274
171274     8888P=*
171274     PSTWO
171275     8888P/
171274     )KILL 8888P
171274
171274  "8BDIS; 8BDIS=0
171274  "
171274
171274
171274
171274  )LINE
```

```
171274  %
171274  % *)9SLPL
171274  )9SLPL171274    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
171274  %                                                                                %
171274  % 50.0 STC MAGTAPE DRIVER     80-07-03   PHE      VERSION  AA                    %
171274  %         LAST MODIFIED       82-02-03   TLS      REV.      01                   %
171274  %         LAST MODIFIED       82-14-06   TP       BIT 6 TO BIT 3 IN STAKO        %
171274  %         LAST MODIFIED       83-05-31   HKD      READ BACKWARDS (CORE.ADDR)     %
171274  %         LAST MODIFIED       83-10-11   TP       ERROR IN STARE CORRECTED       %
171274  %                                                                                %
171274  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
171274  +"99SM1+99SM2
"171274  %                                                                                %
171274  %      CALLING SEQUENCE:                                                         %
171274  %             JPL I (SMAGT                                                       %
171274  %             JMP ERROR       % ERROR EXIT                                       %
171274  %             JMP BUSY        % BUSY EXIT                                        %
171274  %             JMP FINIS       % FINISH EXIT                                      %
171274  %                                                                                %
171274  %      CALLED WITH:                                                              %
171274  %             X-REG:          BUSY INDICATOR WHEN CALLING THE DRIVER             %
171274  %                             OR RECORD SIZE FOR READ/WRITE TYPE COMMANDS        %
171274  %             T-REG:          BITS 0-5: FUNCTION CODE:                           %
171274  %                                       0= READ ONE RECORD                       %
171274  %                                       1= WRITE ONE RECORD                      %
171274  %                                       2= READ ONE RECORD WITH ODD NUMBER OF    %
171274  %                                          BYTES.ITISTHE SAME AS CODE 0 BUT      %
171274  %                                          IT DOES NOT USE ERROR EXIT IF THE     %
171274  %                                          RECORD HAS ODD NUMBER OF BYTES        %
171274  %                                       3= LOOP WRITE TO READ IN FCU             %
171274  %                                       4= READ ONE RECORD BACKWARDS             %
171274  %                                       10=ADVANCE TO EOF (FORWARD SPACE A       %
171274  %                                          TAPEMARK)                             %
171274  %                                       11=REVERSE TO EOF (BACKSPACE A           %
171274  %                                          TAPEMARK)                             %
171274  %                                       12=WRITE EOF (WRITE A TAPEMARK)          %
171274  %                                       13=REWIND TO BOT                         %
171274  %                                       14=ERASE GAP (SAME AS WRITE SKIP)        %
171274  %                                       15=BACKSPACE NO. OF RECORDS              %
171274  %                                       16=FORWARD SPACE NO. OF RECORDS          %
171274  %                                       17=REWIND AND UNLOAD                     %
171274  %                                       20=READ STATUS (COMPATIBLE WITH PERTEC   %
171274  %                                          MAGTAPES)                             %
171274  %                                       21=CLEAR TAPE SUBSYSTEM                  %
171274  %                                       22=CLEAR TAPE SUBSYSTEM WITH ERROR       %
171274  %                                          EXIT                                  %
171274  %                                       24=READ LAST STATUS                      %
171274  %                                       25=READ TAPE STATUS                      %
171274  %                                       26=READ BYTE-RECORD                      %
171274  %                                       27=WRITE BYTE-RECORD                     %
171274  %                                       33=CLEAR SELECTED UNIT                   %
171274  %                                       34=SET DIAGNOSTIC MODE                   %
171274  %                                       35=NO OPERATION                          %
171274  %                                       37=SENSE DRIVE STATUS                    %
171274  %                             BITS 6-8: UNIT NUMBER                              %
171274  %                             BITS 9-15 SHOULD BE 0                              %
171274  %             D-REG:          BITS 0-1: DENSITY                                  %
171274  %                                       0=1600 BPI (PE-MODULATION)               %
171274  %                                       1=6250 BPI (GCR-MODULATION)              %
171274  %                                       2= 800 BPI (NRZI.NOT AVAILABLE           %
```

```
171274   %                                        ON THE STC 1951 DRIVE)                                %
171274   %                                BITS 2-15 SHOULD BE 0                                         %
171274   %                                                                                              %
171274   %                THE 24-BIT MEMORY ADDRESS IS FOUND IN THE DATAFIELD                           %
171274   %                WORDS MEMA1 AND MEMA2 (BITS 0-15 IN MEMA2)                                    %
171274   %                                                                                              %
171274   %                THE 24-BIT WORD-COUNTER IS FOUND IN THE DATAFIELD                             %
171274   %                WORDS DWONO AND CXRG (BITS 0-15 IN CXRG)                                      %
171274   %                                                                                              %
171274   %          RETURN INFORMATION:                                                                 %
171274   %                ERROR EXIT:      X=HARDWARE STATUS (COMPATIBLE WITH OTHER                     %
171274   %                                   MAGTAPES)                                                  %
171274   %                BUSY EXIT:       THE ROUTINE MUST BE CALLED AGAIN AT ONCE OR                  %
171274   %                                 AFTER INTERRUPT WITH PARAMETERS UNCHANGED                    %
171274   %                                 IF FUNCTION WAE STARTED,X=0                                  %
171274   %                                 IF FUNCTION WAS NOT STARTED,X=CONTENT WHEN DRIVER            %
171274   %                                 WAS ENTERED.(X><0)                                           %
171274   %                FINISHED EXIT:   X=HARDWARE STATUS (COMPATIBLE WITH OTHER                     %
171274   %                                   MAGTAPES)                                                  %
171274   %                                 A=CORE ADDRESS LOWER 16 BITS                                 %
171274   %                                 FULL CORE ADDRESS IS FOUND IN DATAFIELD WORDS                %
171274   %                                 CMAD1 AND CMAD2 (BITS 0-15 IN CMAD2)                         %
171274   %                                                                                              %
171274   %                FCU INTERFACE STATUS IS FOUND IN DATAFIELD ARRAY FCST                         %
171274   %                                                                                              %
171274   %          B IS NOT CHANGED                                                                    %
171274   %                                                                                              %
171274   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
171274
171274   %          GLOBAL PARAMETER FIELD FOR STC MAGTAPE DRIVER
171274   INTEGER DEVTL                    % IOX INSTRUCTION
171275   INTEGER XSTAT                    % STATUS WORD RETURNED IN X-REG
171276   INTEGER DENSI                    % DENSITY CODE
171277   INTEGER FUNCO                    % HARDWARE FUNCTION CODE
171300   %INTEGER WONO                     % WORD NUMBER IN DMA
171300   INTEGER UNINO                    % UNIT NO TO BE USED AS ADDRESSING INDEX
171301   INTEGER COCOD                    % COMMAND TYPE CODE. BIT 3 INDICATES FORWARD MOVEM.
171302   %                                % 0=RESET
171302   %                                % 1=DMA READ
171302   %                                % 2=DMA WRITE
171302   %                                % 3=SPACE TYPE COMMAND
171302   %                                % 4=CONTROLLING TYPE COMMANDS
171302   INTEGER ASAV,DSAV,TSAV,XSAV,LSAV
171307
171307   %=============================================================================
171307   % 50.1        S M A G T   E R R E T   B U R E T   F I N E T   E R R E X
171307   %
171307   SUBR SMAGT,ERRET,BURET,FINET,ERREX
171307
171307   SYMBOL SENS1=4,COMW,SENS2,LWCNT
171307
171307   SMAGT: A=:ASAV:=L=:LSAV:=D=:DSAV        % SAVE REGISTERS
171314          T=:TSAV; A:=T/\77=:FUNCO; X=:XSAV
171321
171321          A:=TSAV/\300 SHZ 6=:NEWUN       % INITIALIZE PARAMETERS
171325          A:=TSAV/\300 SHZ -6=:UNINO
171331          A:=DSAV/\3 SHZ 16=:DENSI
171335          A:=HDEV+164000=:DEVTL
171340
171340          IF FUNCO=21 OR =22 THEN         % CLEAR TAPE SUBSYSTEM
```

```
171347            -1=:SELUN
171351            A:=60; T:=DEVTL+DCONT; *EXR ST
171355            CALL STARE; CALL STAKO; T=:XSTAT
171360            GO FAR FINET
171361          FI
171361
171361          IF NEWUN><SELUN THEN              % SELECT ANOTHER UNIT
171365            NEWUN=:SELUN+DENSI
171370            T:=DEVTL+COMW; *EXR ST
171373            A:=4; T-2; *EXR ST
171376          FI
171376          X:=UNINO                          % GET UNIT NUMBER
171377          IF XSAV=0 THEN                    % IF BUSY OR REWINDING ENTRY
171401               COMCO(X)=:COCOD
171403               CALL STACH                   % CHECK FOR ACCEPTABLE STATUS
171404               CALL ERRET
171405               GO XBURET                    % RETRY-COUNTING!!!
171406               GO FAR FINET
171407               *)FILL                       % FOR MAC.
171433          FI
171433          0=:COCOD
171434          CALL STARE                        % READ STATUS OF NEW DRIVE
171435          IF FCST(1) BIT 2 THEN             % SET BOT BIT IN DENSITY CODE
171441            ADNSTY(UNINO) BONE 17=:ADNSTY(X)
171445          ELSE
171446            ADNSTY(UNINO) BZERO 17=:ADNSTY(X)
171452          FI
171452          FCST(1)/\12020-12000
171456          IF A=0 THEN                       % DRIVE READY FOR ANY OPERATION?
171457            X:=XNOWUNIT
171460            0=:X.NRDYF; CALL COMEX
171462          FI
171462          IF FUNCO>17 AND ><26 AND ><27 THEN % FUNCTIONS TO BE EXECUTED
171474                                            % INDEPENDENT OF REWINDING OR ON-
171474                                            % LINE STATUS.
171474            CALL COMEX
171475          FI
171475    XBURET:
171475          FCST(1)/\2000-2000                % TEST FOR DRIVE REWIDING.IF SO,WAIT FOR
171501          IF A=0 THEN                       % 2 MINUTES AND GIVE ERROR MESSAGE.IF NOT
171502            X:=XNOWUNIT
171503            IF X.NRDYF=0 THEN              % REWINDING GIVE ERROR EXIT AT ONCE.
171505              RTRES=:X.NRDYF; -137=:X.NRDTR % FIRST TIME REWINDING
171511              GO FAR BURET
171512            ELSE
171513              IF X.NRDTR><0 THEN
171515                A+1=:X.NRDTR; GO FAR BURET % FOLLOWING TIMES REWINDING
171520              FI
171520            FI
171520          FI
171520          CALL STAKO                        % LAST TIME NOT READY.ERROR EXIT
171521          T=:XSTAT
171522          CALL ERRET
171523    RBUS
171537
171537    %=============================================================================
171537    % 50.2        E R R E T   B U R E T   F I N E T   E R R E X
171537    %
171537    SUBR ERRET,BURET,FINET,ERREX
171537    INTEGER SAVL,CPAR1,CPAR2,CPAR3
```

```
171543    ERRET: A:=20; T:=DEVTL+DCONT; *EXR ST  % ERROR EXIT.CLEAR INTERFACE AND
171547           A:=4000+DENSI+SELUN; T+2; *EXR ST % DRIVE.
171554           A:=4; T-2; *EXR ST
171557    ERREX: A:=L=:SAVL
171561           CPAR1+1=:CPAR1
171564           DSAV=:D:=LSAV=:L:=CMAD2
171571           T:=TSAV; X:=XSTAT
171573           EXIT
171574
171574    BURET: X:=UNINO                          % STORE COMMAND CODE
171575           COCOD=:COMCO(X)
171577           X:=XSAV                          % BUSY EXIT
171600           CPAR2+1=:CPAR2
171603           DSAV=:D:=LSAV+1=:L:=ASAV
171611           T:=TSAV
171612           EXIT
171613
171613    FINET: DSAV=:D:=LSAV+2=:L                % FINISHED EXIT
171620           CPAR3+1=:CPAR3
171623           A:=CMAD2                         % GET CORE ADDRESS REG.
171624           T:=TSAV; X:=XSTAT
171626           EXIT
171627    RBUS
171642
171642    %-----------------------------------------------------------------
171642    %       SUBR COMEX
171642    %       EXECUTES COMMAND AS THIS:
171642    %       CHECKS THE FUNCTION CODE FOR VALIDITY
171642    %       SETS PROPER TYPE CODE IN COCOD
171642    %       MAPS FUNCTION CODE INTO TAPE SYSTEM FORMAT
171642    %       ASSEMBLES COMMAND AND CONTROL WORDS
171642    %       INITIATES FUNCTION
171642    %       RETURNS TO SMAGT ON THE PROPER ENTRY POINT
171642
171642    SYMBOL ILL=-1
171642    SYMBOL RDF=1011
171642    SYMBOL WRT=3012
171642    SYMBOL LWR=7002
171642    SYMBOL RDB=5001
171642    SYMBOL FSF=2413
171642    SYMBOL BSF=403
171642    SYMBOL WTM=1414
171642    SYMBOL REW=3404
171642    SYMBOL ERG=5414
171642    SYMBOL BSB=4403
171642    SYMBOL FSB=6413
171642    SYMBOL RUN=7404
171642    SYMBOL CLR=4004
171642    SYMBOL DMS=2004
171642    SYMBOL NOP=4
171642    SYMBOL SNS=6004
171642
171642    %=================================================================
171642    % 50.3        F U N I L
171642    %
171642    SUBR FUNIL
171642
171642    FUNIL: 60=:XSTAT; CALL ERRET
171645    RBUS
171647
```

```
171647   %=============================================================================
171647   % 50.4        C O M E A
171647   %
171647   SUBR COMEA
171647
171647   INTEGER ARRAY DAMAP:=(RDF,WRT,RDF,LWR,RDB,ILL,ILL,ILL)
171657
171657   COMEA: X:=FUNCO; A:=DAMAP(X)+1
171662          IF A=0 THEN GO FAR FUNIL FI      % CHECK FOR ILLEGAL FUNCTION CODE
171664          A:=DAMAP(X)/\17=:COCOD
171667          IF X=1 THEN
171672            IF FCST(1) BIT 3 THEN
171676              21=:XSTAT; CALL ERRET        % WRITE PROTECT VIOLATION
171701            FI
171701          FI
171701
171701          A:=20; T:=DEVTL+DCONT; *EXR ST   % CLEAR INTERFACE
171705          DWONO; T + 4; *EXR ST            % SET WORD COUNTER BITS 16-23
171710          A:=CXRG; *EXR ST           % SET WORD COUNTER BITS 0-15
171712          A:=MEMA1; T-6; *EXR ST           % WRITE MEMORY ADDRESS REGISTER
171715          A:=MEMA2; *EXR ST
171717          X:=FUNCO; A:=DAMAP(X)/\7400+DENSI+SELUN
171724          T+4; *EXR ST                     % SET COMMAND REGISTER
171726          A:=7; T-2; *EXR ST               % ACTIVATE AND ENABLE INTERRUPTS
171731          0=:XSAV; GO FAR BURET                    % BUSY EXIT
171733   RBUS
171747
171747   %=============================================================================
171747   % 50.5        C O M E B
171747   %
171747   SUBR COMEB
171747   INTEGER ARRAY SPMAP:=(FSF,BSF,WTM,REW,ERG,BSB,FSB,RUN)
171757
171757   COMEB: X:=FUNCO
171760
171760          IF X=12 OR X=14 THEN
171766            IF FCST(1) BIT 3 THEN
171772              21=:XSTAT; CALL ERRET
171775            FI
171775          FI
171775          A:=FUNCO/\7=:X
172000          A:=SPMAP(X)/\17=:COCOD
172003          A:=20; T:=DEVTL+DCONT; *EXR ST   % CLEAR INTERFACE
172007          A:=SPMAP(X)/\7400+DENSI+SELUN    % SET COMMAND REGISTER
172013          T+2; *EXR ST
172015          A:=7; T-2; *EXR ST               % ACTIVATE AND ENABLE INTERRUPTS
172020          0=:XSAV; GO FAR BURET            % BUSY EXIT
172022   RBUS
172036
172036   %=============================================================================
172036   % 50.6        C O M E C
172036   %
172036   SUBR COMEC
172036   COMEC: X:=FUNCO
172037          IF X=26 OR X=27 THEN
172045            CXRG+1 SHZ-1=:CXRG             % CHANGE BYTE COUNTER TO WORD
172051            IF X=26 THEN 2=:FUNCO          % CHANGE FUNCTION CODE
172056            ELSE 1=:FUNCO FI
172061            GO FAR COMEA                   % GOTO READ/WRITE ROUTINE
172062          FI
```

```
172062
172062          4=:COCOD
172064          IF X=20 OR X=24 THEN                % READ STATUS
172072            IF X=20 THEN CALL STARE FI
172076            CALL STAKO; T=:XSTAT              % CONVERT TO X-STATUS
172100            GO FAR FINET                      % FINISHED EXIT
172101          FI
172101          GO FAR FUNIL                        % REST ARE ILLEGAL
172102  RBUS
172112
172112  %=============================================================================
172112  % 50.7         C O M E D
172112  %
172112  SUBR COMED
172112  SYMBOL COMW=5
172112  INTEGER ARRAY REMAP:=(ILL,ILL,ILL,ILL,DMS,NOP,ILL,SNS)
172122
172122  COMED:  X:=FUNCO
172123          4=:COCOD
172125          IF X=33 THEN                        % CLEAR SELECTED DRIVE
172130            CALL STARE; CALL STAKO
172132            T=:XSTAT
172133            A:=20; T:=DEVTL+DCONT; *EXR ST
172137            A:=CLR+SELUN+DENSI; T+2; *EXR ST
172144            A:=4; T-2; *EXR ST
172147            IF FUNCO=22 THEN CALL ERREX
172154            ELSE GO FAR FINET FI
172156          FI
172156          X-30; A:=REMAP(X)
172160          IF A+1=0 THEN GO FAR FUNIL FI       % ILLEGAL FUNCTION CODE
172163          A:=REMAP(X)/\7400+SELUN+DENSI
172167          T:=DEVTL+COMW; *EXR ST              % SET COMMAND REGISTER
172172          A:=4; T-2; *EXR ST                  % ACTIVATE AND ENABLE INTERRUPTS
172175          GO FAR FINET                        % FINISHED EXIT
172176  RBUS
172213
172213  %=============================================================================
172213  % 50.8         C O M E X
172213  %
172213  SUBR COMEX
172213
172213  COMEX:  A:=FUNCO/\70 SHZ -3                 % GO TO ROUTINE FOR PROPER COMMAND TYPE
172216          GOSW FAR COMEA,COMEB,COMEC,COMED,FUNIL,FUNIL,FUNIL,FUNIL
172227  RBUS
172236
172236  %=============================================================================
172236  % 50.9         S T A R E
172236  %
172236  SUBR STARE
172236  %       READS STATUS OF INTERFACE INTO FIRST 4 LOCATIONS OF ARRAY FCST(15)
172236  %       IN THE DATA FIELD POINTED TO BY B-REG
172236
172236  INTEGER SAVL
172237
172237  SYMBOL COMW=5
172237
172237  STARE:  A:=L=:SAVL
172241
172241          T:=DEVTL; *EXR ST                   % GET ENDING CORE ADDRESS
172243          A=:CMAD2
```

```
172244          *EXR ST
172245          A=:CMAD1
172246          T+DST; *EXR ST                  % READ STATUS
172250          A=:FCST(0)
172252          T+2; *EXR ST                    % READ SENSE 1
172254          A=:FCST(1)
172256          T+1; A:=FCST(0)/\177400; *EXR ST
172263          T+1; *EXR ST                    % READ SENSE 2
172265          A/\1777=:FCST(2)
172270          A:=FCST(0)/\177400+1; T-1; *EXR ST
172276          T+1; *EXR ST                    % READ SENSE 3
172300          A/\1777=:FCST(3)
172303          A:=FCST(0)/\177400=:D+3; T-1; *EXR ST; AAT 1; EXR ST
172314          A/\1777=:FCST(4)
172317          FOR X:=5 TO 14 DO               % READ MUX BUS (8 BYTES)
172323             A:=D+X+3; T:=DEVTL+COMW; *EXR ST
172331             T+1; *EXR ST
172333             A/\1777=:FCST(X)
172335          OD
172337          A:=SAVL=:P
172341    RBUS
172344
172344    %=============================================================================
172344    % 50.10       S T A K O
172344    %
172344    SUBR STAKO
172344    %       ASSEMBLES THE STATUS WANTED RETURNED IN THE X-REGISTER                ...
172344    %       STATUS,SENSE 1 AND SENSE 2 ARE USED
172344    %       X-STATUS IS RETURNED IN XSTAT AND T-REGISTER
172344
172344    INTEGER SAVL
172345
172345    STAKO:  A:=L=:SAVL
172347          A:=FCST(0)/\20=:T                 % ERROR
172353          A:=FCST(0)
172355          IF A BIT 6 THEN T BONE 10 FI      % ODD BYTES READ
172360          IF A NBIT 4 AND A NBIT 3 THEN
172364             T BONE 13 FI                   % UNIT BUSY
172365          A:=FCST(1)
172367          IF A BIT 4 THEN T BONE 15 FI      % SET REWINDING STATUS
172372          IF A BIT 1 THEN T BONE 11 FI      % END OF TAPE STATUS
172375          IF A NBIT 3 THEN T BONE 1 FI      % WRITE ENABLE RING PRESENT
172400          IF A BIT 14 AND A BIT 12 THEN T BONE "0" FI  % DRIVE READY STATUS
172405          A/\204; T\/A                      % EOF/BOT STATUS SET
172407          A:=FCST(2)
172411          IF A BIT 3 THEN T BONE 6 FI       % DATA ERROR
172414          IF A BIT "0" AND A BIT 1 THEN     % FATAL ERROR; I.E.
172420             T BONE 3 FI                    % OPERATION INCOMPLETE/REJECTED (CHANGED BY TP 14/6-82)
172421          A/\344                            %   GIVES DATA ERROR
172422          IF A><0 THEN T BONE 12 FI         % TRANSFER ERROR(OR OF 4 BITS)
172424          A:=SAVL=:P
172426    RBUS
172431
172431    %=============================================================================
172431    % 50.11       S T A C H
172431    %
172431    SUBR STACH
172431    %       CHECKS STATUS OF TAPE SUBSYSTEM WHEN BUSY ENTRY.
172431    %       IF STILL NOT READY FOR TRANSFER IT EXITS BUSY AGAIN
172431    %       OTHERWISE IT EXITS EITHER FINISHED OR ERROR
```

```
172431
172431    INTEGER SAVL
172432    INTEGER POINTER PSAVL=?                  % DEFINED LATER!!!
172432    INTEGER CORP1=?,CORP2=?
172432    STACH: A:=L=:SAVL
172434           CALL STARE
172435           CALL STAKO; T=:XSTAT
172437
172437    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% RETURN ACTUAL DENSITY WHEN STARTING FROM BOT.
172437
172437           IF COCOD BIT 3 AND ADNSTY(UNINO) BIT 17 THEN
172446             IF FCST(1) BIT 13 THEN 1=:ADNSTY(UNINO); GO STACI FI
172456             IF FCST(1) BIT 11 THEN 2=:ADNSTY(UNINO); GO STACI FI
172466             0=:ADNSTY(UNINO)
172470           FI
172470    STACI: IF FCST(1) BIT 17 THEN A:=SAVL+1=:P FI      % STILL BUSY EXIT
172477           IF T BIT 7 AND COCOD/\7=1 THEN  % CHECK FOR EOF WHEN READING
172506             XSTAT BONE 4=:XSTAT
172511           FI
172511           IF T BIT 11 AND COCOD BIT 3 THEN % EOT REACHED WHEN A FORWARD
172516             XSTAT BONE 4=:XSTAT            % MOVING COMMAND IS TRIED.
172521           FI
172521           IF FUNCO=0 OR A=4 THEN
172526             IF XSTAT BIT 10 THEN
172531               XSTAT BONE 4=:XSTAT          % ODD NUMBER OF BYTES IN RECORD WITH
172534             FI
172534           FI
172534
172534           GO CONT1
172535
172535    *)FILL
172544
172544    %%% READ AND CHECK ENDING MEMORY ADRESS
172544
172544    INTEGER CORP1,CORP2
172546
172546    CONT1: IF CTRG/\77=4 GO CONT2              % READ BACKWARDS
172553           IF COCOD/\7=2 OR =1 THEN            % IF DMA,CHECK ENDING MEM.ADRESS.
172563             MEMA2+CXRG=:CORP2                 % CALCULATE EXPECTED ENDING MEMORY
172566             MEMA1; *RADD ADC DA               % ADDRESS.
172570             A+DWONO=:CORP1
172572             IF COCOD/\7 =1 THEN               % IF READ-TYPE COMMAND AND
172577               IF XSTAT NBIT 7 THEN            % IF NOT EOF STATUS
172602                 IF CMAD1>>CORP1 THEN          % CHECK IF ENDING MEMORY ADR. TOO
172606                   XSTAT\/4020=:XSTAT          % LARGE
172611                 ELSE
172612                   IF CMAD2>>CORP2 AND CMAD1=CORP1 THEN
172622                     XSTAT\/4020=:XSTAT
172625                   FI
172625                 FI
172625               FI
172625             ELSE
172626               IF XSTAT NBIT 6 THEN                      % IF OUTPUT AND NOT BAD TAPE
172631                 IF CORP2><CMAD2 OR CORP1><CMAD1 THEN    % CHECK FOR MEMORY ADDRESS NOT
172641                   XSTAT\/4020=:XSTAT                    % EQUAL TO EXPECTED.
172644                 FI
172644               FI
172644             FI
172644           FI
172644
```

```
172644   CONT2: IF COCOD/\7=1 THEN                % OVERFLOW IN READ TEST
172651            IF FCST(3) NBIT 7 THEN
172655              XSTAT\/10020/\177377=:XSTAT
172661            FI
172661          FI
172661          IF FUNCO=13 THEN                  % SIMULATE CORRECT LOAD-POINT STATUS
172665            XSTAT/\20007\/5=:XSTAT          % AFTER REWIND COMMAND IS ISSUED.
172671          FI
172671          IF FUNCO=17 THEN                  % LEGAL TO BE NOT READY AFTER THESE
172675            XSTAT/\2100
172677            IF A=0 THEN A:=PSAVL+2=:P FI
172703          FI
172703          IF XSTAT BIT 4 THEN
172706            A:=PSAVL=:P                      % OTHER ERRORS
172710          FI
172710          A:=PSAVL+2=:P                      % NO ERRORS,ACCEPTABLE STATUS.HURRA!
172713
172713   INTEGER POINTER PSAVL := SAVL             % DUE TO 'RANGE EXCEEDED'!!!
172714
172714   RBUS
172727   *)KILL ASAV DSAV TSAV XSAV LSAV
172727
172727   *"
"172727    *)9SCLC

172727  %
172727  % @EOF
172727  %
```

```
172727  )LINE
172727  %=============================================================================
172727  % 47.0        B U F F E R   S I Z E S
172727  %
172727  *  8S3C
172730  BP00L7
030330  *)
030331  )MCDEF ACE9I $SIZB
99EIO=9EIOB; )KILL 9EIOB; 9EIOB=99EIO+$SIZB ; )KILL 99EIO
)
030331
030331  %=============================================================================
030331  % 47.1        I O B U T
030331  %
030331  9EIOB=9EMRE                                  % START OF I/O BUFFERS
030331  IOBUT=*
030331  "8SM01+NSBM1
030331          20006 ;MIBU1
030333          120006;MOBU1
030335  "IND01;200    ;IDBUS
030337          ACE9I IDBUS
030337  99EIO=9EIOB; )KILL 9EIOB; 9EIOB=99EIO+IDBUS; )KILL 99EIO
030337
030337  "IND02;201    ;IDBUS
030341          ACE9I IDBUS
030341  99EIO=9EIOB; )KILL 9EIOB; 9EIOB=99EIO+IDBUS; )KILL 99EIO
030341
030341  "
030341          500    ;100050
030343          ACE9I 50
030343  99EIO=9EIOB; )KILL 9EIOB; 9EIOB=99EIO+50; )KILL 99EIO
030343
030343  "8BCH1;1237   ;300
030345          ACE9I 300
030345  99EIO=9EIOB; )KILL 9EIOB; 9EIOB=99EIO+300; )KILL 99EIO
030345
030345  "8C1HD; 41360;4
030347        141361;4
030351          ACE9I 10
030351  99EIO=9EIOB; )KILL 9EIOB; 9EIOB=99EIO+10; )KILL 99EIO
030351
030351  "8C2HD;41362;4
030353         141363;4
030355          ACE9I 10
030355  99EIO=9EIOB; )KILL 9EIOB; 9EIOB=99EIO+10; )KILL 99EIO
030355
030355  "
030355  -1
030356  EIOBU=*
030356
030356  )LINE
030356  "8BDIS+8BDIM+8ZBDI
```

```
030356
030356
030356  %=============================================================================
030356  % 43.5        B D I S K                    VERSION A       26/11-81    CORR. BY TP
030356  %                                          VERSION B       30/12-81    CORR. BY TP
030356  %                                          VERSION C.6     16/ 4-82    CORR. BY TP
030356  %                                          VERSION D        6/10-82    CORR. BY TP
030356  %                                          VERSION E       10/11-82    CORR. BY TP
030356  %                                          VERSION E.1     29/ 3-83    CORR. BY TP
030356  %                                          VERSION E.2     31/ 8-83    CORR. BY TP
030356  %                                          VERSION E.3     26/ 9-83    CORR. BY TP
030356  %                                          VERSION E.4     19/ 3-84    CORR. BY TP
030356  %                                          VERSION F.0      2/ 4-84    CORR. BY TP
030356  %                                          VERSION F.1      3/ 5-84    CORR. BY TP
030356  %                                          VERSION F.2     26/ 6-84    CORR. BY TP
030356  %                                          VERSION F.3     31/ 7-84    CORR. BY TP
030356  %                                          VERSION F.4      3/ 9-84    CORR. BY TP
030356  %
030356  %    SUPER DISK (AND ECC CARTRIDGE DISKS) TRANSFER ROUTINE
030356  %
030356  %          THERE ARE 3 VERSIONS OF THE DRIVER, ACCORDING TO LIBRARY MARKS:
030356  %
030356  %                              8BDIS = NORMAL SINTRAN DRIVER
030356  %                              8BDIM = TEST PROGRAM DRIVER
030356  %                              8ZBDI = SWAP DRIVER (READ ONLY)
030356  %
030356  %       8BDIS AND 8BDIM
030356  %
030356  %          THE ARRAY ',B HTABL' MUST BE CORRECT (SEE BELOW).
030356  %
030356  %       8BDIS
030356  %
030356  %          THE DISC-DESCRIBING TABLES ARE EXTERNAL (SEE BELOW).
030356  %
030356  %       8ZBDI
030356  %
030356  %          THE CORRECT DEVICE DESCRIPTOR IS GENERATED AT ASSEMBLY TIME!!!
030356  %          PLDWO AND PSTWO WILL NOT WORK FOR N10.
030356  %
030356  %       FOR ALL:
030356  %
030356  %          IF THE ROUTINES 'PLDWO' AND 'PSTWO' ARE UNDEFINED, THEY WILL
030356  %          BE INCLUDED AFTER THE DRIVER!!!
030356  %          FOR N10, PAGING MUST BE OFF WHEN PLDWO AND PSTWO ARE ENTERED
030356  %          (USED BY ERROR CORRECTION). NOT FOR 8ZBDI. SEE 8ZBDI ABOVE.
030356  %          ERROR CORRECTION ASSUMES THAT THE CPU CAN REACH ALL THE MEMORY
030356  %          ACCESSIBLE TO THE DISK.
030356  %
030356  %
030356  "8ZBDI -8BDIS
030356  %
030356  %       CALL:
030356  %       JPL  I (ZBDIS
030356  %       JMP    ERROR            %ERROR EXIT
030356  %       JMP    BUSY             % BUSY EXIT
030356  %       JMP    FINIS            %FINISHED EXIT
030356  %
030356  "8ZBDI+8BDIS+8BDIM
030356  %
030356  %       REGISTER CONTENTS WHEN ROUTINE IS CALLED
```

```
030356  %
030356  %       T-  BIT 0-5:  DEVICE OPERATION  (FOR 8ZBDI, READ ONLY)
030356  %                      0 READ TRANSFER
030356  %                      1 WRITE TRANSFER
030356  %                      2 READ PARITY TRANSFER
030356  %                      3 COMPARE TRANSFER
030356  %                     20 READ LAST STATUS
030356  %                     35 RELEASE                (NOT FOR THE SWAP DRIVER)
030356  %                     36 PRIORITY-SELECT        (NOT FOR THE SWAP DRIVER)
030356  %                     37 THIS FUNCTION BELONGS TO FUNCTION 42 AND IS STARTED
030356  %                        BY THE DRIVER. IT SHOULD NOT BE USED IN A CALL.
030356  %                     42 TEST IF SPARE TRACKS AND IF SPARE SECTOR FORMAT.
030356  %                        (NOT FOR THE SWAP DRIVER).
030356  %                        IT IS ASSUMED THAT NO OF SECTORS IN X IS 1 AND THAT
030356  %                        SECTOR NUMBER (LOGICAL DISK ADDR) IN AD IS 0 (EXCEPT
030356  %                        THE FIXED BIT IN D 017) WHEN THIS FUNCTION CALLS THE
030356  %                        DRIVER THE FIRST TIME.
030356  %                     43 READ  FORMAT TABLE (NOT FOR THE SWAP DRIVER)
030356  %                     44 WRITE FORMAT TABLE (NOT FOR THE SWAP DRIVER)
030356  %          BIT 6-8:   UNIT SELECT
030356  %          BIT 9-11:          SURFACE NUMBER FOR PHOENIX DISK
030356  %          BIT 12-13: CORE ADDRESS BITS 16-17 (N10)
030356  %          BIT 14:    DEVICE TYPE
030356  %                             1= 30/60/90 MBYTE
030356  %                             0= ANY OTHER TYPE
030356  %
030356  %          BIT 15:    0: NORMAL TRANSFER
030356  %                     1: BIT 6 IS ADDED TO THE SECTOR NUMBER
030356  %                        (USED TO READ SPARE TRACKS. ONLY POSSIBLE FOR
030356  %                        THE LIBRARY MARK 8BDIM).
030356  %
030356  %       A- LOGICAL ADDRESS (DISK SECTOR NUMBER ) BITS 16-31
030356  %
030356  %       D- LOGICAL ADDRESS (DISK SECTOR NUMBER ) BITS 0-15
030356  %                          LOGICAL ADDRESS BIT 15 MEANS ANY FIXED
030356  %                          SURFACE FOR CARTRIDGE DISK
030356  %
030356  %       X- NUMBER OF SECTORS TO BE TRANSFERED
030356  %
030356  %         ****   ,B MEMAD CONTAINS THE 24-BIT MEMORY ADRESS
030356  %                         IN A DOUBLE WORD.
030356  %         ****   ,B HDEV  CONTAINS HARDWARE DEVICE NUMBER
030356  %
030356  "8ZBDI+8BDIS+8BDIM
030356  %
030356  %        DISC LAYOUT DEFINITION:
030356  %
030356  %                                WORD 0 (BSECW) = WORDS/SECTOR
030356  %                                     1 (BSECT) = SECTORS/TRACK
030356  %                                     2 (BSECY) = SECTORS/CYLINDER
030356  %                                     3 (BMXCY) = MAXIMUM VALUE OF CYLINDER
030356  %                                     4           FIRST CYL OF SPARE TRACK AREA
030356  %                                     5           FORMAT IN ECC CONTROL WORD
030356  %                                     6           FIRST CYL OF EXTRA AREA
030356  %                                     7           POINTER TO RELATED LAYOUT
030356  %
030356  %       EXIT INFORMATION:
030356  %
030356  %       ERROR EXIT: X- HARDWARE STATUS REG.
030356  %                   T- BIT  9: SEEK ERROR. RETURN TO ZERO SEEK HAS BEEN DON
```

```
030356  %                   BIT 10: CORE ADDR. REG NOT AS EXPECTED
030356  %                   BIT 11: LOG. BLOCK ADDR. OUT OF RANGE, OR
030356  %                           TRANSFER BIGGER THAN 63.5K
030356  %                   BIT 12: DATA CORRECTION INFORMATION ERROR
030356  %                   BIT 13: DRIVE TYPE DEFINITION ERROR
030356  %                   BIT 14: ILL. DEVICE OP. CODE
030356  %                   BIT 15: CONTROLLER NOT ACTIVE AFTER ACTIVATE
030356  %     EVERY TIME THE DRIVER ERROR EXITS, IT STORES 11 WORDS AT THE END OF
030356  %     THE CODE. THESE WORDS, THEREFORE, WILL CONTAIN SIGNIFICANT INFORMATION
030356  %     ABOUT THE LAST ERROR THAT OCCURED. THE ADDRESS OF THIS FIELD CAN BE
030356  %     FOUND IN BDISK+2.
030356  %
030356  %     BUSY EXIT:      THE ROUTINE MUST BE CALLED AGAIN AT ONCE, OR
030356  %                     AFTER INTERRUPT. THE NEW CALL IS DONE WITH T,A,D AND
030356  %                     X-REG. UNCHANGED.
030356  %
030356  %     FINISHED EXIT: X-HARDWARE STATUS REG.
030356  %                     WITH FUNCTION 042, X WILL BE:
030356  %                     0: NOT SPARE TRACKS, NOT SPARE SECTOR FORMAT
030356  %                     1:     SPARE TRACKS, NOT SPARE SECTOR FORMAT
030356  %                     2: NOT SPARE TRACKS,     SPARE SECTOR FORMAT
030356  %                     3:     SPARE TRACKS,     SPARE SECTOR FORMAT
030356  %
030356  %     B-REG. IS NOT CHANGED BY THE ROUTINE
030356  %
030356  %       THE ROUTINE CONVERTS LOGICAL (SECTOR NUMBER) ADDR. (LA)
030356  %       TO HARDWARE (CYLINDER/HEAD/SECTOR) ADDR. (HA)
030356  %       ACCORDING TO TABLE HTABL FOR THE CORRESPONDING UNIT.
030356  %
030356  %     **** THE NUMBER OF SECTORS FOR THE VARIOUS DISK PACKS, AND THE ACTUAL TABLE NAMES:
030356  %
030356  %          37 MB =  110176                    ; DT037
030356  %          70 MB =  220526                    ; DT070
030356  %          75 MB =  220526                    ; DT075
030356  %         135 MB =  421206                    ; DT135   SPARE SECTOR FORMAT
030356  %         140 MB =  441254                    ; DT140
030356  %         160 MB =  441254                    ; DT160
030356  %         288 MB = 1045672                    ; DT288
030356  %         285 MB = 1066450                    ; DT285   SPARE SECTOR FORMAT
030356  %         300 MB = 1127720                    ; DT300
030356  %         450 MB = 1542420                    ; DT450   SPARE SECTOR FORMAT
030356  %         460 MB = 1603760                    ; DT460
030356  %
030356  %          30 MB = 0034736 FOR EACH SURFACE   ; DT030
030356  %          60 MB = 0034736 FOR EACH SURFACE   ; DT030
030356  %          90 MB = 0034736 FOR EACH SURFACE   ; DT030
030356  %
030356  %
030356  %        WHEN CALLING, THE LOGICAL ADDR. IN AD MUST
030356  %        BE REPRESENTED AS AN UNSIGNED MAGNITUDE NUMBER.
030356  %
030356  %
030356  %        THE SECTORS WITHIN THE LAST CYLINDER ARE TRANSFERED
030356  %        FIRST, THEN THE SECTORS WITHIN THE PREVIOUS CYLINDER ETC....
030356  %        FOR THE OLD INTERFACE (10 KHZ), WITHIN ONE CYLINDER,
030356  %        THE GREATEST SINGLE TRANSFER IS  64K-BSECW, WHERE BSECW IS NUMBER
030356  %        OF WORDS PER SECTOR (THE WORD COUNT REGISTER HAS ONLY 16 BITS).
030356  %        FOR THE NEW INTERFACE (15KHZ), STILL WITHIN ONE CYLINDER,
030356  %        THE GREATEST SINGLE TRANSFER CAN BE A WHOLE CYLINDER, SINCE
030356  %        THE WORD COUNT REGISTER HAS 24 BITS.
```

```
030356  %
030356  %      DISPLACEMENTS FOR IOX INSTRUCTIONS:
030356  %
030356           RCA=0                  % READ CORE ADDRESS
030356           LCA=1                  % LOAD CORE ADDRESS
030356           RSC=2                  % READ SEEK CONDITION / ECC COUNT
030356           LBA=3                  % LOAD BLOCK ADDRESS  I/II
030356           RSR=4                  % READ STATUS REG / ECC PATTERN
030356           LCO=5                  % LOAD CONTROL WORD
030356           LWC=7                  % LOAD WORD COUNT / ECC CONTROL
030356        INIOX=165540              % INITIAL IOX INSTRUCTION
030356
030356  % THE DISK DRIVER MAY BE CALLED WITH 11 DIFFERENT DEVICE OPERATIONS.
030356  % IN THE FOLLOWING, READ, READ PARITY, AND COMPARE, ARE CALLED NON-WRITE.
030356  % THE WRITE OPERATION IS CALLED WRITE. THE READ-STATUS OPERATION IS QUITE
030356  % SPECIAL. IT IS REALLY A READ-LAST-STATUS OPERATION, AS IT WILL READ
030356  % LAST STATUS FROM THE VARIABLE SSTAT. IT WILL ALSO INITIATE THE DISK DATA
030356  % FIELD VARIABLES BUSFL, ECCFL, MARGC, AND SMARG BY STORING ZERO INTO THEM,
030356  % AND BY SETTING ERRC1 TO -4 AND ERRC2 TO -28.
030356  % THE TEST-IF-SPARE-TRACK OPERATION WILL BE EXECUTED FIRST AS A PARITY READ
030356  % FROM THE LAST POSSIBLE SECTOR ON THE FIRST TRACK, AND THEN AS A PARITY READ
030356  % FROM THE FIRST SECTOR IN THE SPARE TRACK AREA. IT IS NOT FOR THE SWAP DRIVER.
030356  % RELEASE AND PRIORITY-SELECT ARE NOT FOR THE SWAP DRIVER EITHER. THEY ARE
030356  % USED WHEN MORE THAN ONE CPU USE THE DISK.
030356  %
030356  % ***** THE READ-STATUS OPERATION SHOULD NOT BE USED WHEN BUSFL IS NONZERO.****
030356  %
030356  % IF THE DISK DRIVER IS CALLED WITH READ-ZERO-SECTORS, THE STATUS REGISTER
030356  % WILL BE READ AND PUT IN THE X REGISTER, AND THE DISK DRIVER WILL EXIT
030356  % AT THE FINISHED EXIT. IF NOT-ON-CYLINDER (STATUS BIT 14),
030356  % THE DISK DRIVER WILL EXIT AT THE ERROR EXIT.
030356  %
030356  %
030356  % THE VARIABLES IN THE DISK DATA FIELD USED BY THE DISK DRIVER:
030356  %
030356  % ARG
030356  %
030356  % USED TO SAVE THE VALUE OF THE A-REGISTER AT ENTRY.
030356  %
030356  % BUSFL
030356  %
030356  % BUSY FLAG. CAN HAVE THE VALUES ZERO OR NONZERO.
030356  % MUST BE ZERO THE FIRST TIME THE DISK DRIVER IS CALLED. THE DISK DRIVER
030356  % WILL SET BUSFL TO NONZERO WHEN A TRANSFER IS STARTED, AND WILL SET IT BACK
030356  % TO ZERO WHEN A TRANSFER IS FINISHED, OR WHEN IT ERROR EXITS.
030356  %
030356  % CDISP
030356  %
030356  % USED FOR ECC COUNT AND SECTOR WORD NUMBER.
030356  % WHEN THE DISK DRIVER TRIES TO DO ERROR CORRECTION ON A SECTOR, IT READS
030356  % THE ECC COUNT AND STORES IT IN CDISP. IF 1 < CDISP < 8204, IT IS CHANGED
030356  % INTO A SECTOR WORD NUMBER IN THE RANGE 0-0777, POINTING TO A DOUBLE WORD
030356  % IN THE SECTOR THAT MAY BE CHANGED, DEPENDING ON THE ECC PATTERN (SEE
030356  % CPAT1 AND CPAT2). IF ERROR CORRECTION HAS TAKEN PLACE (SEE CORCU), THEN
030356  % IT IS POSSIBLE FOR THE CALLING PROGRAM TO KNOW WHERE THE CORRECTION TOOK
030356  % PLACE, BY INSPECTING CDISP.
030356  %
030356  % CMAD1
030356  %
030356  % POINTS TO THE FIRST WORD OF A DOUBLE WORD CONTAINING CURRENT MEMORY ADDRESS.
```

```
030356  % THE CURRENT MEMORY ADDRESS IS THE MEMORY ADDRESS USED BY THE DRIVER
030356  % WHEN IT STARTS A TRANSFER. IT IS NOT NECESSARILY EQUAL TO THE MEMORY
030356  % ADDRESS SUPPLIED IN THE CALL (SEE MEMA1, MEMA2).
030356  % IF THE DISK DRIVER BREAKS A TRANSFER INTO SMALLER TRANSFERS, FOR INSTANCE,
030356  % IT HAS TO CALCULATE A NEW MEMORY ADDRESS AND STORE IT IN CMAD1 AND CMAD2.
030356  %
030356  % CMAD2
030356  %
030356  % POINTS TO THE SECOND WORD OF A DOUBLE MEMORY ADDRESS. SEE CMAD1.
030356  %
030356  % CORCU
030356  %
030356  % CORRECTION COUNTER. EVERY TIME THAT ERROR CORRECTION OCCURS,
030356  % CORCU IS INCREMENTED BY 1. ERROR CORRECTION IS ALWAYS DONE ON SINGLE SECTORS.
030356  % THE NOT-WRITE RETRY COUNTER (SEE SRTRY) IS USUALLY ALSO INCREMENTED (ONCE
030356  % FOR EACH SECTOR IN THE TRANSFER) WHEN CORCU IS INCREMENTED.
030356  % CORCU IS INCREMENTED ON READ AND PARITY-CHECK OPERATIONS.
030356  % THE FAILING BITS CAN BE IN THE DATA PART AND/OR IN THE ECC POLYNOMIAL.
030356  %
030356  % CPAT1
030356  %
030356  % WHEN THE DISK DRIVER WANTS TO DO ERROR CORRECTION, IT READS THE ELEVEN BITS
030356  % OF THE ECC PATTERN, INVERTS THE BIT ORDER (SWAPS BIT 0 AND 10, BIT 1 AND 9,
030356  % ETC.), AND STORES IT TEMPORARILY IN CPAT1. CPAT1 SHOULD NOT BE ZERO.
030356  % THEN CPAT1 IS SHIFTED ACCORDING TO THE LEAST SIGNIFICANT BITS OF THE ECC
030356  % COUNT IN CDISP. THE RESULT IS STORED IN CPAT1 AND CPAT2, READY TO BE
030356  % EXCLUSIVE OR'ED INTO THE SECTOR JUST READ.
030356  %
030356  % CPAT2
030356  %
030356  % SEE CPAT1.
030356  %
030356  % DRG
030356  %
030356  % USED TO SAVE THE VALUE OF THE D-REGISTER AT ENTRY.
030356  %
030356  % ECCFL
030356  %
030356  % RUN-ECC OPERATION FLAG. CAN HAVE THE VALUES ZERO AND NONZERO.
030356  % THIS FLAG IS USUALLY ZERO, BUT WHEN THE DISK DRIVER WANTS TO DO ERROR
030356  % CORRECTION, IT SETS ECCFL TO NONZERO AND STARTS THE RUN-ECC OPERATION.
030356  % WHEN RUN-ECC IS FINISHED, ECCFL IS SET BACK TO ZERO.
030356  % ECCFL IS SET TO ZERO AT THE ERROR EXIT AND THE FINISHED EXIT.
030356  %
030356  % ERRC1
030356  %
030356  % ERROR COUNTER. SHOULD HAVE A NEGATIVE VALUE. BEFORE THE DISK DRIVER IS CALLED
030356  % THE FIRST TIME, ERRC1 USUALLY SHOULD HAVE THE VALUE -4. A FAILING WRITE
030356  % TRANSFER WILL THEN BE RETRIED THREE TIMES. IF THE CALLING PROGRAM WANTS
030356  % NOT TO HAVE WRITE RETRIES, IT SHOULD STORE -1 IN ERRC1 BEFORE CALLING
030356  % THE DISK DRIVER. FOR FAILING NOT-WRITE TRANSFERS, ERRC1 SHOULD BE HANDLED
030356  % IN THE SAME WAY, BUT THEN ERRC2 MUST ALSO BE TAKEN INTO ACCOUNT.
030356  % ERRC1 IS SET TO -4 WHEN A TRANSFER IS FINISHED, WHEN THE DISK DRIVER IS
030356  % IN ERROR RECOVERY MODE (SEE SMARG), WHEN ALL WRITE-RETRIES HAVE BEEN DONE,
030356  % OR WHEN ALL NOT-WRITE MARGINAL RECOVERIES HAVE BEEN DONE (SEE MARGC).
030356  %
030356  % ERRC2
030356  %
030356  % MARGINAL RECOVERY COUNTER. SHOULD HAVE A NEGATIVE VALUE. BEFORE THE DISK
030356  % DRIVER IS CALLED THE FIRST TIME, ERRC2 USUALLY SHOULD HAVE THE VALUE -28.
```

```
030356  % A FAILING NOT-WRITE TRANSFER WILL FIRST DO SOME RETRIES ACCORDING TO THE
030356  % VALUE OF ERRC1. WHEN ERRC1 HAS BEEN COUNTED UP TO ZERO, THERE WILL BE NO
030356  % MORE ORDINARY RETRIES. INSTEAD, THE DISK DRIVER WILL TRY TO DO MARGINAL
030356  % RECOVERY CYCLES, ACCORDING TO THE VALUE OF ERRC2. IT WILL FIRST SET MARGC
030356  % NONZERO. USUALLY, THE DISK DRIVER WILL DO 27 MARGINAL RECOVERIES (THREE
030356  % FULL CYCLES). BUT IF THE CALLING PROGRAM SETS BOTH ERRC1 AND ERRC2 TO -1,
030356  % THE DISK DRIVER WILL DO NO ORDINARY RETRIES AND NO MARGINAL RECOVERY CYCLES.
030356  % ERRC2 IS SET TO -28 WHEN A TRANSFER IS FINISHED, WHEN THE DISK DRIVER IS IN
030356  % ERROR-RECOVERY MODE (SEE SMARG), WHEN ALL WRITE-RETRIES HAVE BEEN DONE,
030356  % OR WHEN ALL NOT-WRITE MARGINAL RECOVERIES HAVE BEEN DONE (SEE MARGC).
030356  %
030356  % HDEV
030356  %
030356  % CONTAINS HARDWARE DEVICE NUMBER (01540 OR 01550).
030356  %
030356  % HTABL
030356  %
030356  % THERE ARE FOUR WORDS IN THE DATAFIELD (ONE FOR EACH UNIT), EACH POINTING TO
030356  % AN EIGHT-WORD DISK LAYOUT VECTOR. HTABL POINTS TO THE FIRST OF THE FOUR DATA
030356  % FIELD WORDS.
030356  %
030356  % MARGC
030356  %
030356  % MARGINAL RECOVERY FLAG. CAN HAVE THE VALUES ZERO AND NONZERO.
030356  % WHEN ALL THE ORDINARY NOT-WRITE RETRIES HAVE BEEN DONE (IF A TRANSFER FAILS),
030356  % THE MARGINAL RECOVERY BIT IN THE CONTROL WORD IS TURNED ON, MARGC IS SET
030356  % TO NONZERO, AND TYPEC IS INCREMENTED IN BIT 9.
030356  % MARGC IS SET TO ZERO WHEN THE DISK DRIVER IS IN ERROR RECOVERY
030356  % MODE (SEE SMARG), AND WHEN A TRANSFER IS FINISHED (BOTH OK EXIT AND ERROR
030356  % EXIT).
030356  %
030356  % MEMA1
030356  %
030356  % THE MOST SIGNIFICANT PART OF A DOUBLE WORD CONTAINING MEMORY ADDRESS.
030356  %
030356  % MEMA2
030356  %
030356  % THE LEAST SIGNIFICANT PART OF THE MEMORY ADDRESS (SEE MEMA1).
030356  %
030356  % NWLBA
030356  %
030356  % NEW SURFACE AND SECTOR. USED WHEN SPARE TRACK ALLOCATION IS ACTIVE.
030356  % A TRACK THAT IS FLAGGED AS A BAD TRACK, CONTAINS THE ADDRESS OF THE
030356  % RESERVE TRACK IN EVERY DOUBLE WORD. THE FIRST PART OF THE DOUBLE WORD
030356  % CONTAINS THE NEW CYLINDER NUMBER, AND THE SECOND PART CONTAINS THE NEW
030356  % SURFACE NUMBER IN THE LEFTMOST 8 BITS, AND THE SECTOR NUMBER IN THE
030356  % RIGHTMOST 8 BITS. THIS SECTOR NUMBER IS IRRELEVANT, AS THE OLD SECTOR
030356  % NUMBER (THE SECTOR NUMBER IN THE ORIGINAL DISK TRANSFER) IS USED INSTEAD.
030356  % NWLBA MUST BE NWLBB+1. (*****)
030356  %
030356  % NWLBB
030356  %
030356  % NEW CYLINDER NUMBER. SEE NWLBA. MUST BE NWLBA-1 !!
030356  %
030356  % OCMD1
030356  %
030356  % OLD COMPUTED MEMORY ADDRESS, BITS 23-16. USED TO SAVE THE MEMORY ADDRESS
030356  % OF A DISK TRANSFER DURING SPARE TRACK ALLOCATION.
030356  %
030356  % OCMD2
```

```
030356  %
030356  % OLD COMPUTED MEMORY ADDRESS, BITS 15-0. SEE OCMD1.
030356  %
030356  % OSVBA
030356  %
030356  % OLD SURFACE AND SECTOR. USED TO SAVE THE OLD VALUES OF THE DISK ADDRESS
030356  % DURING SPARE TRACK ALLOCATION.
030356  %
030356  % OSVBB
030356  %
030356  % OLD CYLINDER. SEE OSVBA.
030356  %
030356  % OSVCO
030356  %
030356  % OLD CONTROL WORD. USED TO SAVE THE OLD CONTROL WORD DURING SPARE TRACK
030356  % ALLOCATION.
030356  %
030356  % OSVWC
030356  %
030356  % OLD WORD COUNT, LEAST SIGNIFICANT 16 BITS.
030356  % USED TO SAVE THE OLD WORD COUNT DURING SPARE TRACK ALLOCATION.
030356  %
030356  % OSVWK
030356  %
030356  % OLD WORD COUNT, MOST SIGNIFICANT 8 BITS.
030356  % USED TO SAVE THE OLD WORD COUNT DURING SPARE TRACK ALLOCATION.
030356  %
030356  % SCADR
030356  %
030356  % THE LEAST SIGNIFICANT PART OF THE EXPECTED CORE ADDRESS REGISTER WHEN
030356  % A TRANSFER HAS FINISHED. IT IS COMPUTED BY THE DISK DRIVER (CMAD2+SVLWC).
030356  %
030356  % SLONG
030356  %
030356  % COUNTER FOR ERROR-RECOVERY (SEE SMARG). WHEN THE DISK DRIVER INITIATES
030356  % ERROR-RECOVERY MODE, SLONG IS SET TO 1-N, WHERE N IS THE NUMBER OF SECTORS
030356  % IN THE TRANSFER.
030356  %
030356  % SMARG
030356  %
030356  % ERROR-RECOVERY FLAG. CAN HAVE THE VALUES ZERO OR NONZERO.
030356  % WHEN A NOT-WRITE TRANSFER FAILS, THE DISK DRIVER INITIATES ERROR-RECOVERY
030356  % BY SETTING SMARG TO NONZERO. ERROR-RECOVERY MEANS THAT THE TRANSFER IS
030356  % BROKEN INTO SINGLE-SECTOR TRANSFERS. EACH OF THESE TRANSFERS IS THEN DONE
030356  % AND CHECKED FOR ERRORS. IF THE TRANSFER IS READ, AND IT FAILS, ERROR
030356  % CORRECTION IS TRIED. IF THIS FAILS, OR IF THE TRANSFER IS PARITY CHECK OR
030356  % COMPARE, THE USUAL RETRIES AND MARGINAL RECOVERIES ARE DONE.
030356  % SMARG IS SET TO ZERO WHEN A NEW TRANSFER IS STARTED, AND WHEN A TRANSFER
030356  % IS FINISHED (BOTH OK EXIT AND ERROR EXIT).
030356  %
030356  % SPACO
030356  %
030356  % SPARE TRACK ALLOCATION COUNTER. EVERY TIME A DISK TRANSFER HAS SUCCESSFULLY
030356  % USED A RESERVE TRACK, THIS COUNTER IS INCREMENTED BY ONE.
030356  %
030356  % SPAFL
030356  %
030356  % SPARE TRACK ALLOCATION FLAG.
030356  % 0:  NORMAL TRANSFER (NO SPARE TRACKS INVOLVED).
030356  % 1:  ADDRESS MISMATCH HAS OCCURED. SPARE TRACK ADDRESS-READ HAS BEEN STARTED.
```

```
030356  % 2:  TRANSFER IN THE SPARE TRACK AREA HAS BEEN STARTED.
030356  %
030356  % SRTRY
030356  %
030356  % NOT-WRITE RETRY COUNTER. IT IS INCREMENTED EVERY TIME AN ORDINARY NOT-WRITE
030356  % RETRY IS STARTED. MARGINAL RECOVERY CYCLE RETRIES ARE COUNTED IN TYPEC,
030356  % BITS 15-9, EVERY TIME MARGC IS SET NONZERO.
030356  %
030356  % SVLBA
030356  %
030356  % SAVED DISK ADDRESS (SURFACE AND SECTOR).
030356  %
030356  % SVLBB
030356  %
030356  % SAVED DISK ADDRESS (CYLINDER NUMBER).
030356  %
030356  % SVLCA
030356  %
030356  % SAVED LEAST SIGNIFICANT PART OF THE CURRENT MEMORY ADDRESS (EQUAL TO CMAD2).
030356  %
030356  % SVLCO
030356  %
030356  % SAVED CONTROL WORD, EXCEPT WHEN RUN-ECC OPERATION IS STARTED.
030356  %
030356  % SVLWC
030356  %
030356  % SAVED WORD COUNT, LEAST SIGNIFICANT 16 BITS (NUMBER OF SECTORS * BSECW).
030356  %
030356  % SVLWK
030356  %
030356  % SAVED WORD COUNT, MOST SIGNIFICANT 8 BITS (NUMBER OF SECTORS * BSECW).
030356  %
030356  % SWTRY
030356  %
030356  % WRITE RETRY COUNTER. SWTRY IS INCREMENTED BY ONE EACH TIME A WRITE TRANSFER
030356  % FAILS.
030356  %
030356  % TRG
030356  %
030356  % USED TO SAVE THE VALUE OF THE T-REGISTER AT ENTRY.
030356  %
030356  % TYPEC
030356  %
030356  % BIT    0:    1: ECC INTERFACE                    0: 33/66MB INTERFACE
030356  %        1:    1: HEAD ADVANCE                        0: NOT HEAD ADVANCE
030356  %        2:    1: ERROR CORRECTION NOT ALLOWED        0: ERROR CORRECTION ALLOWED
030356  %        3:    1: MARGINAL RECOVERY NOT ALLOWED       0: MARGINAL RECOVERY ALLOWED
030356  %        4:    1: HEAD ADVANCE NOT ALLOWED, UNIT 0    0: HEAD ADVANCE ALLOWED, UNIT 0
030356  %                 (SPARE TRACK ALLOC. ALLOWED)             (NO SPARE TRACK ALLOCATION)
030356  %        5:    1: DITTO FOR UNIT 1                    0: DITTO FOR UNIT 1
030356  %        6:    1: DITTO FOR UNIT 2                    0: DITTO FOR UNIT 2
030356  %        7:    1: DITTO FOR UNIT 3                    0: DITTO FOR UNIT 3
030356  %        8:    1: AUTOMATIC RELEASE NOT ALLOWED       0: ALLOWED
030356  %        9:    1: NEW (15 KHZ) INTERFACE              0: OLD (10 KHZ) INTERFACE
030356  %    15-10:    MARG. REC. COUNTER (INCREMENTED EACH TIME MARGC IS SET NONZERO)
030356  %
030356  %     BITS 0 AND 1 ARE SET INSIDE THE DISK DRIVER.
030356  %     BITS 2, 3, AND 4-7 ARE SET BY THE CALLING PROGRAM.
030356  %          BITS 4-7 CAN ALSO BE SET BY THE FUNCTION TEST-IF-SPARE-TRACK (042)
030356  %     BIT  8 IS SET BY THE FUNCTION PRIORITY-SELECT, AND RESET BY THE FUNCTION
```

```
030356  %           RELEASE. IT MAY ALSO BE SET/RESET BY THE CALLING PROGRAM.
030356  %    BIT  9 IS SET INSIDE THE DISK DRIVER.
030356  %
030356  % XRG
030356  %
030356  % USED TO SAVE THE VALUE OF THE X-REGISTER AT ENTRY.
030356  %
030356
030356  "8ZBDI -8BDIS
030356
030356  )KILL BSECW BSECT BSECY BMXCY
030356
030356  ZBDIS=*
030356          STF    TRG ,B
030357          STX    XRG ,B
030360          JMP    MORE9
030361
030361  QDLAY=*
030361
030361  "/B100 8ZBDI -8BDIS
030361
030361  BSECW, 1000                    % WORDS/SECTOR                          OLD 75MB
030362  BSECT, 22                      % SECTORS/TRACK
030363  BSECY, 132                     % SECTORS/CYLINDER
030364  BMXCY, 1466                    % MAX CYLINDER VALUE
030365          1465                   % FIRST SPARE TRACK CYLINDER
030366          0                      % FORMAT IN ECC CONTROL                                  ...
030367          0                      % FIRST CYLINDER OF EXTRA AREA
030370          0
030371
030371  "8ZBDI -8BDIS
030371
030371  MORE9=*
030371
030371  "8ZBDI+8BDIS+8BDIM
030371
030371          JMP    BRCHK
030372
030372  )FILL
030372
030372  BRCHK, COPY   SL DA            % SAVE L
030373          STA I (9LREG
030374          SAA    77
030375          AND    TRG ,B
030376
030376  "8ZBDI -8BDIS
030376
030376          JAZ    L41             % THE ONLY LEGAL OPCODE IS 0
030377
030377  "8BDIS+8BDIM+8ZBDI
030377
030377          JMP I (ILCOD           % ILL. DEVICE OP. CODE
030400  L41,    LDA    TRG ,B
030401          AND    (0700           % MASK FOR UNIT ADDRESS BITS.
030402          SHA    ZIN  1          % SHIFT TO POSITION FOR CWR.
030403          STA I (BTSTA           % STORE VALUE
030404          LDA    SVLCO,B         % GET LAST COMMAND
030405          AND    (16000          % MASK FOR WRITE AND COMPARE BIT, AND MARG.REC. BIT
030406          ADD I (BTSTA           % ADD TO NEW COMMAND IN ORDER NOT TO
030407          STA I (BTSTA           % REVERSE THE FIFO DIRECTION IN BLCOG
```

```
030410          LDA    BUSFL,B          % PREVIOUS X-FER TO BE CHECKED?
030411          JAF    BRSR1            % NO - ,CONTINUE
030412          STZ    SPAFL ,B         % RESET SPARE TRACK ALLOCATION FLAG
030413          LDA I  (BTSTA
030414  BLCOG,  INIOX  LCO              % SET CWR15=0
030415  BRSR1,  INIOX  RSR              % READ STATUS
030416          BSKP   ZRO 20 DA        % CONTROLLER ACTIV?
030417  L39,    JMP I  (EXBUS           % YES, BUSY EXIT
030420          LDA    BUSFL,B          % PREVIOUS TRANSFER TO BE CHECKED?
030421          JAF    BRSR2            % YES. SKIP UNIT SELECT
030422          LDA    TRG ,B
030423          AND    (0700
030424          SHA    ZIN 1
030425          BSET   ONE 30 DA
030426          JPL    BLCO1            % SELECT UNIT  (TEST MODE)
030427          AAA    010
030430          JPL    BLCO1            % CLEAR DEVICE
030431  BRSR2,  INIOX  RSR              % READ STATUS
030432          STA I  (SSTAT           % SAVE IT
030433          BSKP   ONE 160 DA       % ON CYLINDER?
030434          JMP I  (ERCYL           % NO. ERROR
030435          LDA    BUSFL,B
030436          JAZ    L18              % ZERO: NO PREVIOUS TRANSFER. JUMP TO BCONT
030437          JMP    I *+1            % JUMP TO SPARE TRACK ALLOCATION HANDLER
030440          SPHAN
030441  L33,    LDA    SMARG,B          % ERROR RECOVERY CYCLE ?
030442          JAF    SECPR            % NONZERO: YES. JUMP TO SECTOR PROCESSING
030443          LDA I  (SSTAT           % TEST STATUS
030444          AND    (037760
030445          JAF    BREX             % NONZERO: ERRORS IN THE PREVIOUS TRANSFER
030446          JPL I  (BRSR6           % READ AND CHECK THE CORE ADDR REG
030447          JMP I  (ECADR
030450  L18,    JMP I  (BCONT           % NO ERRORS, CONTINUE
030451
030451  BREX=*
030451
030451  "8BDIS+8BDIM+8ZBDI
030451
030451          LDA    TYPEC ,B         % IS ERROR CORRECTION ALLOWED ?
030452          BSKP   ZRO 020 DA
030453          JMP I  (BRTRY           % NO
030454
030454  % PREPARE ERROR RECOVERY BY INITIATING SECTOR-BY-SECTOR PROCESSING (NOT WRITE)
030454
030454          SAA    -1
030455          STA    SMARG,B          % SET SECTOR RECOVERY FLAG
030456          LDD    SVLWC-1 ,B       % GET LEAST WORD COUNT TO D-REG
030457          LDA    SVLWK ,B         % MOST SIGN WORD COUNT
030460          LDX I  (BSECW
030461          RDIV   SX               % CONVERT TO NUMBER OF SECTORS
030462          LDX    XRG ,B
030463          COPY   SA DA CM2
030464          COPY   SA DA AD1
030465          STA    SLONG,B          % STORE SECTOR CONTROL NUMBER
030466          LDA I  (BSECW
030467          STA    SVLWC,B          % SET WORD COUNT TO ONE SECTOR
030470          STZ    SVLWK ,B
030471          JMP I  (BRTRY
030472
030472  "8BDIM+8BDIS+8ZBDI
```

```
030472
030472  )FILL
030510
030510  BLCO1,  INIOX LCO               % LOAD CONTROL WORD
030511          EXIT
030512
030512  % ERROR RECOVERY (SECTOR-BY-SECTOR PROCESSING)
030512  %
030512  SECPR,  LDA I (SSTAT            % GET STATUS
030513          AND   (037760
030514          JAF   ERRAN             % NONZERO: ERROR DURING PREVIOUS TRANSFER
030515  L4,     JPL I (BRSR6            % READ AND CHECK THE CORE ADDR REG
030516          JMP I (ECADR
030517          LDA   SLONG,B           % TEST SECTOR RETRY NUMBER
030520          JAP   L18               % POS: SECTOR RETRY COMPLETED. JUMP TO BCONT
030521          SAA   037
030522          AND   SVLBA ,B          % MASK OUT SECTOR BITS
030523          SUB I (BSECT            % SECTORS/TRACK
030524          AAA   1                 % FIRST=0 NOT 1!!
030525          JAF   L10               % JUMP IF NOT LAST SECTOR ON TRACK
030526          LDA   SVLBA,B
030527          AND   (17400            % MASK FOR HEAD BITS
030530          ADD   (400              % INCR. HEAD ADDR.
030531          STA   SVLBA,B
030532          JMP   L11
030533  L10,    MIN   SVLBA,B           % INCREMENT SECTOR ADDRESS
030534  L11,    MIN   SLONG,B           % INCREMENT SECTOR CONTROL NUMBER
030535          0
030536          LDA   CMAD2,B
030537          ADD I (BSECW            % ADD ONE SECTOR
030540          STA   SVLCA,B           % NEW CORE ADDR.     (16 LOWER)
030541          STA   CMAD2,B
030542          LDA   CMAD1,B
030543          COPY  SA DA ADC
030544          STA   CMAD1,B           % NEW CORE ADDR.    (BANK BITS)
030545          JPL I (BQERR            % SUCCESSFULL SECTOR RETRY. RESET ERROR COUNTERS
030546          STZ   MARGC,B           % RESET MARGINAL CYCLE FLAG
030547          LDA   SVLCO,B
030550          BSET  ZRO 120 DA
030551          STA   SVLCO,B           % RESET COMMAND WORD
030552          AND   (017600
030553          STA I (BTSTA
030554          JMP I (BRTRY
```

```
030555
030555  % PREPARE FOR ERROR CORRECTION (CHECK IF CONDITIONS ARE FAVORABLE)
030555  %
030555  ERRAN=*
030555
030555  "8BDIS+8BDIM+8ZBDI
030555
030555  L24,    LDA I (SSTAT            % GET STATUS
030556          BSKP  ONE 110 DA        % SKIP IF DATA ERROR
030557          JMP   RETRY             % IF NOT --RETRY.
030560          AND   (036740           % MASK FOR ALL ERRORS (OTHERS)
030561          JAF   RETRY             % JUMP TO RETRY IF ANY ERRORS
030562  BRSC5,  INIOX RSC               % READ SEEK CONDITION
030563          BSKP  ZRO 170 DA        % SKIP IF NOT ADDRESS FIELD
030564          JMP   RETRY             % ADDRESS FIELD,-GO TO RETRY
030565          COPY  SA DT             % SAVE SEEK CONDITION
030566          LDA   ECCFL,B           % IS ECC OP. FLAG SET?
030567          JAZ   RECOP             % JUMP IF FLAG NOT SET
030570          BSKP  ZRO 150 DT        % SKIP IF MAX ECC COUNT
030571          JMP   ECCOP             % CORRECTABLE !
030572          STZ   ECCFL,B           % PREPARE FOR RETRY
030573          LDA I (BTSTA            %
030574          BSET  ONE 170 DA
030575          JPL   BLC01             % SET CWR15=1.
030576  BRSC3,  INIOX RSC               % READ ECC COUNT
030577          SUB   (20071            % COUNT SHOULD BE =8249
030600          JAF   ERREC
030601  RETRY,  JPL I (RESEC            % RESET ECC
030602          JMP I (BRTRY            % MARGINAL RECOVERY CYCLE EXIT
030603  RECOP,  SAA   -1
030604          STA   ECCFL,B
030605          LDA I (BTSTA
030606          BSET  ZRO 0140 DA       % REMOVE BIT 12 IF PARITY CHECK
030607          ADD   (40005            % SET CWR14=0,ACTIVATE,ENBL.INTRPT.
030610          JPL   BLC01             % START ECC-OP.
030611          JMP I (EXBUS
030612
030612  )FILL
```

```
030633
030633  % ERROR CORRECTION
030633  %
030633  ECCOP,  STZ    ECCFL,B          % ECCFL:=0
030634          LDA I  (BTSTA
030635          BSET   ONE 170 DA
030636          JPL    BLCO1            % SET CWR15=1.
030637  BRSR4,  INIOX  RSR              % READ ECC PATTERN
030640          SAT    -13
030641          STT    CDISP ,B
030642          SAT    0
030643          JMP    L2               % START PATTERN MIRRORING
030644  L1,     SHA    SHR 1
030645          SHT    ZIN 1
030646  L2,     BSKP   ZRO 0 DA
030647          BSET   ONE 0 DT
030650          MIN    CDISP ,B
030651          JMP    L1
030652          COPY   ST DA
030653          STA    CPAT1,B          % STORE PATTERN TEMPORARILY
030654          JAZ    ERREC
030655  BRSC4,  INIOX  RSC              % READ ECC COUNT ( 2-020001 FOR BIT IN SECTOR)
030656          JAN    ERREC
030657          STA    CDISP ,B         % SAVE ECC COUNT
030660          AAA    -2
030661          JAN    ERREC            % NEG: COUNT < 2
030662          SUB    K8191
030663          AAA    -071
030664          JAP    ERREC            % POS: COUNT > 8249
030665          AAA    056
030666          JAP    AYPAS            % POS: COUNT > 8203.  DATA ALREADY OK
030667          LDA I  (BTSTA
030670          JPL    BLCO1            % SET CWR15=0.
030671          LDA    CDISP ,B         % ECC COUNT
030672          AAA    -014
030673          RCLR   DL               % L:=0
030674          JAP    L6
030675          RINC   DL               % L:=1
030676          AAA    012              % WORDNO MUST NOT BE NEGATIVE
030677  L6,     SAD    SHR 4
030700          STA    CDISP ,B         % WORDNO IN SECTOR
030701          SAA    0
030702          SAD    4
030703          COPY   SA DD
030704          SAA    0105
030705          SKP    DL EQL 0
030706          AAA    012              % BECAUSE OF WORDNO COMPENSATION ABOVE
030707          RSUB   SD DA
030710          ADD    SADIN            % SAD 0
030711          COPY   SA DX
030712          LDA    CPAT1 ,B         % ECC PATTERN
030713          RCLR   DD
030714          EXR    SX               % SHIFT PATTERN
030715          STA    CPAT1 ,B         % SAVE CORRECTION PATTERNS
030716          COPY   SD DA
030717          STA    CPAT2 ,B
030720
030720    8BDIM+8BDIS+8ZBDI
030720
030720          LDD    CMAD1-1 ,B       % CMAD1 TO D
```

```
030721          LDA    CDISP ,B         % WORDNO IN SECTOR
030722          ADD    CMAD2 ,B
030723          SWAP   SA DD
030724          COPY   SA ADC DA
030725          STD    ADSAV            % SAVE CORRECTION ADDR
030726          LDX    CPAT1 ,B         % PATTERN 1
030727          JXZ    L5
030730
030730  "8ZBDI -8BDIS
030730
030730          EXAM                    % GET DATA
030731
030731  "8BDIM+8BDIS+8ZBDI
030731
030731          REXO   SX DT            % CORRECT DATA
030732
030732  "8ZBDI -8BDIS
030732
030732          DEPO
030733
030733  "8BDIM+8BDIS+8ZBDI
030733
030733  L5,     LDD    ADSAV            % 24-BIT ADRESS
030734          RINC   DD
030735          COPY   ADC SA DA
030736          STD    ADSAV            % SAVE NEXT 24 BIT ADRESS
030737          LDX    CDISP ,B         % WORDNO IN SECTOR
030740          AAX    1
030741          BSKP   ZRO 0110 DX
030742          JMP    AYPAS            % WORDNO IS 01000.  DONT CORRECT
030743          LDX    CPAT2 ,B         % PATTERN 2
030744          JXZ    AYPAS
030745
030745  "8ZBDI -8BDIS
030745
030745          EXAM                    % NEXT DATA WORD
030746
030746  "8BDIM+8BDIS+8ZBDI
030746
030746          REXO   SX DT            % CORRECT DATA
030747
030747  "8ZBDI -8BDIS
030747
030747          DEPO
030750
030750  "8BDIM+8BDIS+8ZBDI
030750
030750  AYPAS,  MIN    CORCU ,B         % INCREASE CORRECTION COUNTER
030751          0
030752          LDA I  (SSTAT           % REMOVE DATA ERROR BIT FROM STATUS
030753          AND    (176757
030754          STA I  (SSTAT
030755          LDD    ARG ,B           % RESET AD
030756          LDX    XRG ,B
030757          JPL I  (RESEC           % RESET ECC
030760          JMP I  (L4
030761
030761  ERREC,  JMP I  (ERREB           % JUMP-HELP
030762
030762  KBIBI,  017777
```

```
030763  SADIN, SAD   0
030764  ADSAV, 0; 0
030766
030766  )FILL
030774
030774  % TRY AGAIN AFTER ERROR (NOT WRITE)
030774  %
030774  BRTRY, LDA    MARGC,B          % MARC. REC. CYCLE FLAG ALREADY SET?
030775         JAF    BMARG            % NONZERO; YES
030776         STZ    ECCFL,B          % RESET ECC PROCESSING FLAG
030777         MIN    ERRC1,B          % ORDINARY RETRIES DONE?
031000         JMP    L12              % NO. TRY ONCE MORE
031001         LDA    TYPEC ,B         % MARG REC ALLOWED ?
031002         BSKP   ZRO 030 DA
031003         JMP I  (ERR             % NO
031004         ADD    (02000           % INCREMENT MARG REC COUNTER (BITS 15-10)
031005         STA    TYPEC ,B
031006         SAA    -1
031007         STA    MARGC,B          % SET MARG REC. CYCLE FLAG
031010         LDA    SVLCO,B
031011         BSET   ONE 120 DA       % SET BIT 10 IN CONTROL WORD(MARG. REC.)
031012         STA    SVLCO ,B
031013         AND    (017600
031014         STA I  (BTSTA
031015  BMARG, MIN    ERRC2,B          % ALL MARG. REC.'S DONE?
031016         JMP    BOM
031017         JMP I  (ERR             % 3 MARG. REC. CYCLES DONE. ERROR
031020  L12,   MIN    SRTRY ,B
031021         0
031022  %
031022  % TRANSFER ROUTINE
031022  %
031022  BOM,   LDA I  (BTSTA
031023         BSET   ONE  40 DA       % CLEAR DEVICE    ***NEW,IMPROVED!***
031024         JPL    BLCOF            % CWR 15:=0
031025  BRSR8, INIOX  RSR              % CLEAR UPPER/LOWER CORE ADDR. FLIP-FLOP
031026         LDA    SVLBA,B          % HEAD AND SECTOR
031027
031027  "8BDIS+8BDIM+8ZBDI
031027
031027  BLBA1, INIOX  LBA              % LOAD HEAD AND SECTOR
031030         LDA I  (BTSTA
031031         BSET   ONE 170 DA
031032         JPL    BLCOF            % CWR 15:=1
031033         LDA    SVLBB,B
031034
031034  "8BDIS+8BDIM+8ZBDI
031034
031034  BLBA3, INIOX  LBA              % LOAD CYLINDER ADDRESS
031035         JPL I  (RESEC           % RESET ECC.  SET CWR15:=0
031036         LDA    CMAD1,B
031037  BLCA1, INIOX  LCA              % LOAD BANK NO
031040         LDA    CMAD2,B
031041  BLCA3, INIOX  LCA              % LOAD MEMORY ADDR WITHIN BANK
031042         LDA    SVLWK ,B         % MOST SIGN WORD COUNT
031043  BLWC2, INIOX  LWC
031044         LDA    SVLWC,B
031045  BLWC1, INIOX  LWC              % LOAD WORD COUNT
031046         LDA    SVLCO,B
031047         JPL    BLCOF            % START TRANSFER
```

```
031050  BRSR3,  INIOX RSR               % READ STATUS
031051          STA I (SSTAT
031052          BSKP  ONE 20 DA         % CONTROLLER ACTIVE ?
031053          JMP I (ERACT            % NO
031054          LDA   SVLWC ,B          % WORD COUNT
031055          ADD   CMAD2 ,B          % LEAST SIGN. PART OF MEMORY ADDR
031056          STA   SCADR ,B          % EXP. LEAST SIGN. PART OF THE CORE ADDR REG
031057  EXBUS,  SAA   -1
031060          STA   BUSFL,B           % SET BUSY FLAG
031061          LDA I (9LREG
031062          COPY  SA DL
031063          LDF   TRG ,B
031064          EXIT  AD1               % BUSY EXIT!!!
031065
031065  )FILL
```

```
031075
031075  % START A NEW TRANSFER WHEN X IS NONZERO
031075  %
031075  BCONT, SKP    IF DX UEQ 0
031076         JMP I  (FINEX          % ALL BLOCKS TRANSFERED
031077         STZ    SMARG,B
031100  BRSC6, INIOX  RSC             % READ SEEK COND.
031101         COPY   SA DT
031102         LDA I  (BTSTA          %
031103         BSET   ONE 170 DA
031104         JPL    BLCOF           % SET CWR 15 =1.
031105         LDA    TYPEC ,B
031106         COPY   SA DL
031107         BSET   ONE 0110 DL     % BIT FOR 15KHZ INTERFACE
031110  BRSR5, INIOX  RSR             % READ ECC PATTERN
031111         AND    (174000         % MASK FOR BITS 11-15 OF A-REG.
031112         SUB    (134000
031113         JAZ    NEWIE           % ZERO: 15KHZ INTERFACE
031114         BSET   ZRO 0110 DL
031115         BSET   ZRO 0160 DA
031116         JAZ    NEWIE           % ZERO: 10 KHZ INTERFACE
031117         JMP I  (ERTYP          % ERROR.
031120  %
031120  % DECIDE IF HEAD-ADVANCE SHOULD BE USED
031120  %
031120  NEWIE, COPY   SL DA
031121         STA    TYPEC ,B
031122         COPY   AD1 DL          % L:=1
031123
031123  "8BDIS+8BDIM+8ZBDI
031123
031123  L19,   SAA    -4
031124         AND    TYPEC ,B        % REMOVE BITS 1-0 FROM TYPEC
031125         RADD   SL DA
031126         STA    TYPEC ,B        % 1 OR 3 IN BITS 1-0
031127  %
031127  % COMPUTE THE BLOCK ADDRESS OF THE LAST BLOCK (SECTOR) IN THE TRANSFER
031127  %
031127         LDA    ARG ,B          % MOST SIGN. OF LOG. DISK ADDR
031130         LDT    TRG ,B
031131         BSKP   ONE 0160 DT     % PHOENIX DISK ?
031132         JMP    XY              % NO
031133         BLDA   170 DD          % SAVE FIXED/REMOV INDICATOR IN K
031134         BSET   ZRO 170 DD
031135         SAT    0
031136         BSET   BAC 140 DT      % SURFACE ALWAYS 20+N FOR FIXED DISK
031137         STT    SAVT
031140         JMP    XY
031141
031141  SAVT,  0
031142
031142  *FILL
031147  *7000, 07000
031150
031150  BLCOF, INIOX  LCO             % LOAD CONTROL WORD
031151         EXIT
031152
031152  XY,    AAX    -1
031153         RADD   SX DD
031154         COPY   SA ADC DA       % ADDR OF LAST SECTOR NOW IN AD
```

```
031155          AAX   1
031156          LDT I (BSECY            % SECTORS/CYL
031157          RDIV  ST
031160          LDT I (BMXCY
031161          SKP   DT MGRE SA
031162  L40,    JMP I (ILADR            % CYLINDER > MAX CYLINDER
031163          STA   SVLBB,B           % SAVE CYLINDER ARGUMENT
031164          COPY  SD DA AD1
031165          STA   HESEC             % 0 < HESEC <= BSECY
031166          SKP   DX GRE SA         % CHECK IF TRANSFER IS GREATER THAN 63.5K
031167          COPY  SX DA
031170          LDT I (BSECW            % A IS NO. OF SECTORS IN TRANSFER
031171          RMPY  ST DA
031172          JAZ   L20               % ZERO: LESS THAN 64K
031173          SWAP  SA DD
031174          RADD  ST DA
031175          RADD  ADC CM1 DD
031176          SWAP  SA DD
031177          RDIV  ST                % AFTER THIS, A HOLDS NUMBER OF SURPLUS SECTORS
031200          LDT   TYPEC ,B
031201          BSKP  ZRO 0110 DT
031202          SAA   0                 % 15 KHZ INTERFACE. ALLOW MORE THAN 63.5K
031203  L20,    COPY  SA DL             % SAVE IT IN L FOR A LITTLE WHILE
031204          LDA   HESEC
031205          COPY  SA DD
031206  %
031206  % COMPUTE SURFACE AND SECTOR NUMBER
031206  %
031206          LDA   TYPEC,B
031207          BSKP  ONE 10 DA         % SKIP IF HEAD ADVANCE
031210          JMP   L14
031211          COPY  SX DA
031212          SKP   IF DA LST SD      % SKIP IF WHOLE TRANSFER WITHIN CYLINDER
031213          COPY  SD DA
031214          RSUB  SA DD
031215          RADD  SL DD             % SURPLUS SECTORS
031216          JMP   L23
031217  L14,    RDCR  DD
031220  L23,    SAA   0
031221          LDT I (BSECT            % GET SECTORS/TRACK
031222          RDIV  ST                % A:=SURFACE NO.
031223          COPY  SD DT             % D&T=NUMBER OF SECTORS IN LAST CYLINDER
031224          SHA   ZIN  10           % SURFACE IN LEFT BYTE
031225          RADD  SA DD             % MERGE SURFACE AND SECTOR
031226          AAT   1                 % NO. OF SECTORS IN LAST CYLINDER
031227          LDA   TRG ,B
031230          BSKP  ONE 160 DA        % IF CARTRIDGE DISK
031231          JMP   NOTPH
031232          AND   K7000             % MASK FOR SURFACE (LA BIT 16-18)
031233          SHA   ZIN SHR 1         % MOVE TO BIT 8-10
031234          ADD   SAVT              % ADD SECTOR AND FIXED/REMOVABLE SURFACE FLAG
031235          ADD   HESEC
031236          AAA   -1
031237          COPY  SA DD
031240          AND   (03400
031241          JAZ   NOTPH             % ZERO: SURFACE 0 OR 020
031242          BSKP  ONE 0140 DD
031243          JMP   L40               % REMOVABLE, AND SURFACE # 0 !!
031244  NOTPH,  LDA   TYPEC,B
031245          BSKP  ONE 10 DA         % SKIP IF HEAD ADVANCE
```

```
031246          JMP   L15
031247          COPY  SD DA
031250          JMP   L3
031251  L15,    SKP   IF DX GRE ST
031252          COPY  SX DT             % T: NO. OF SECTORS TO TRANSFER
031253          COPY  SD DA AD1
031254          RSUB  ST DA             % A: HARDWARE BLOCK ADR.
031255  L3,     STA   SVLBA,B           % SAVE HEAD AND SECTOR
031256  %
031256  % DECREMENT X
031256  %
031256          LDA   TYPEC,B
031257          BSKP  ONE 10 DA
031260          JMP   L7                % JUMP IF NOT HEAD ADVANCE
031261          LDT   HESEC             % LOAD HEAD + SECTOR
031262          SKP   IF DX GRE ST
031263          COPY  SX DT
031264          RSUB  SL DT             % SURPLUS SECTORS
031265  L7,     RSUB  ST DX             % X: NO. OF BLOCKS LEFT TO THE NEXT TRANSFER
031266          STX   XRG ,B
031267  %
031267  % X IS NOW NO. OF BLOCKS TO BE TRANSFERRED NEXT TIME.
031267  % T IS NOW NO. OF BLOCKS TO TRANSFER THIS TIME.
031267  % COMPUTE SVLCA, CMAD1, CMAD2, SVLWC.
031267  %
031267          LDA I (BSECW
031270          RMPY  SX DA             % AD: X*BSECW=NO. OF WORDS LEFT
031271          SWAP  SA DD
031272          ADD   MEMA2 ,B
031273          COPY  SD ADC DD
031274          STA   SVLCA ,B
031275          SWAP  SA DD
031276          ADD   MEMA1 ,B
031277          STD   CMAD1 ,B
031300          LDA I (BSECW
031301          RMPY  ST DA             % AD: T*BSECW=NO. OF WORDS TO TRANSFER
031302          STA   SVLWK ,B          % SAVE MOST SIGN WORD COUNT
031303          COPY  SD DA
031304          STA   SVLWC,B           % SAVE WORD COUNT
031305  %
031305  % PREPARE BTSTA AND THE CONTROL WORD (SVLCO)
031305  %
031305  "8BDIS+8BDIM+8ZBDI
031305
031305          LDA   TRG ,B
031306          AND   K0700
031307          SHA   ZIN 1             % UNIT NO
031310
031310  "8BDIS+8BDIM+8ZBDI
031310
031310          STA I (BTSTA
031311          AAA   5                 % ACTIVATE & ENABLE INTERRUPT
031312          STA   SVLCO,B           % SAVE CONTROL WORD
031313
031313  "8BDIS+8BDIM+8ZBDI
031313
031313  L42,    JMP I (BOM              % JUMP TO TRANSFER ROUTINE
031314
031314  )FILL
031324  K0700, 0700
```

```
031325  HESEC, 0
031326
031326  % THIS IS THE FINISHED-AND-OK EXIT
031326  %
031326  FINEX, LDX    SSTAT
031327         JPL    BQERR            % RESET ERROR COUNTERS
031330         STZ    ECCFL,B
031331         STZ    SMARG,B
031332         STZ    MARGC,B
031333         STZ    BUSFL,B
031334         STZ    SPAFL ,B
031335
031335  "8BDIS+8BDIM+8ZBDI
031335
031335         LDA    9LREG
031336         COPY   SA AD1 DL
031337         LDF    TRG ,B
031340         EXIT   AD1              % FINISHED EXIT
031341  %
031341  % RESET-ECC SUBROUTINE
031341  %
031341  RESEC, LDA    BTSTA
031342         BSET   ONE 170 DA
031343  BLCOH, INIOX LCO               % SET CWR15 = 1
031344         SAA    1
031345         ADD I (BMXCY+2
031346  BLWC3, INIOX LWC               % RESET ECC
031347         LDA    BTSTA
031350  BLCOB, INIOX LCO               % SET CWR15=0.
031351         EXIT
031352  %
031352  % READ-AND-CHECK THE CORE ADDR REG
031352  %
031352  BRSR6, INIOX RSR               % CLEAR UPPER/LOWER CORE ADDR. FLIP-FLOP
031353
031353  "8BDIS+8BDIM+8ZBDI
031353
031353  BRCA,  INIOX RCA               % READ CORE ADDR. REG
031354         STA    DRIAR
031355         SUB    SCADR,B
031356         JAF    L37              % NONZERO: CORE ADDR. REG NOT AS EXPECTED
031357         LDA    SVLWC ,B         % WORD COUNT
031360         ADD    CMAD2 ,B         % LEAST MEM ADDR
031361         LDA    CMAD1 ,B         % BANK NO
031362         COPY   SA ADC DA        % EXP. BANK NO
031363         ADD    SVLWK ,B         % MOST SIGN WORD COUNT
031364         STA    SAVBN
031365  BRCA2, INIOX RCA               % READ BANK NO
031366         AND    K0377
031367         SUB    SAVBN
031370         STA    SAVBN            % SAVE RESULT TEMP.
031371  BRSC1, INIOX RSC
031372         BSKP   ONE 0140 DA
031373  L38,   EXIT   AD1              % N10 INTERFACE, OR OK
031374         LDA    SAVBN
031375         JAZ    L38
031376  L37,   EXIT                    % BANK NO, OR ADDR, NOT AS EXPECTED
031377
031377  SAVBN, 0
031400  K0377, 0377
```

```
031401
031401  )FILL
031402
031402  % RESET ERROR COUNTERS
031402
031402  BQERR,  SAA    -4
031403          STA    ERRC1 ,B
031404          SAA    -034
031405          STA    ERRC2 ,B
031406          EXIT
031407          EXIT

031407  % BELOW FOLLOW THE DIFFERENT ERROR EXITS
031407  %
031407  ERACT,  SAT    0100             % NOT ACTIVE AFTER ACTIVATE
031410          JMP    L9               % SET BIT 017 IN T
031411
031411  ILCOD,  SAT    040              % ILLEGAL DEVICE OPERATION
031412          JMP    L9               % SET BIT 016 IN T
031413
031413  ERTYP,  SAT    020              % CONTROLLER DEFINITION PROBLEM
031414          JMP    L9               % SET BIT 015 IN T
031415
031415  ERREB,  SAT    010              % ECC PROBLEM
031416          JMP    L9               % SET BIT 014 IN T
031417
031417  ILADR,  SAT    4                % LOGICAL DISK ADDR IS TOO BIG                    ...
031420          JMP    L9               % SET BIT 013 IN T
031421
031421  ECADR,  SAT    2                % CORE ADDR REG ERROR. SET BIT 012 IN T
031422
031422  L9,     SHT    011              % SHIFT ERROR BIT TO CORRECT PLACE
031423          LDA    TRG ,B
031424          AND    BMASK
031425          RADD   SA DT
031426          SAA    0
031427          JPL    BLC08            % CWR15:=0
031430  BRSC7,  INIOX  RSC
031431          STA    RSCON
031432          JPL    BQERR            % RESET ERROR COUNTERS
031433          JMP    ERREK
031434
031434  ERCYL,  SAX    0                % NOT ON-CYLINDER
031435          JMP    L36              % CONTINUE WITH X=0
031436
031436  ERR,    JPL    BQERR            % ALL ERRC1 (WRITE) OR ERRC2 (NON-WRITE) TRANSFERS ARE DONE
031437          SAX    -1               % SET X TO NONZERO
031440  L36,    LDA    TRG ,B
031441          AND    BMASK
031442          COPY   SA DT
031443  BRSC2,  INIOX  RSC
031444          STA    RSCON
031445          BSKP   ZRO 130 DA
031446          BSET   ONE 110 DT       % SEEK ERROR
031447          BSKP   ONE 130 DA
031450          JXZ    ERREK            % JUMP TO ERREK IF X=0 AND NOT SEEK ERROR
031451          LDA    TRG ,B
031452          AND    K0700
031453          SHA    ZIN 1
031454          AAA    20
```

```
031455  BLCO3, INIOX LCO              % DEVICE CLEAR
031456         ADD   (033775          % RETURN-TO-ZERO SEEK (TEST MODE)
031457         LDX   RSCON
031460         BSKP  ONE 0140 DX
031461         BSET  ZRO 030 DA       % N-10.  REMOVE TEST MODE
031462  BLCO4, INIOX LCO
031463  ERREK, STZ   BUSFL,B
031464         STZ   ECCFL,B
031465         STZ   SMARG,B
031466         STZ   MARGC,B
031467         STZ   SPAFL ,B
031470
031470  "8BDIS+8BDIM+8ZBDI
031470
031470         LDX   SSTAT
031471
031471  "8BDIS+8BDIM+8ZBDI
031471
031471         LDA   ARG ,B
031472         JMP I 9LREG            % EXIT
031473
031473  )FILL
031474
031474  SSTAT, 0
031475  BTSTA, 0
031476  BMASK, 0777
031477  9LREG, 0
031500
031500  % BELOW FOLLOW 11 WORDS THAT CONTAIN ERROR INFORMATION ABOUT THE LAST ERROR.
031500  % THEY ARE STORED THERE WHEN THE DRIVER ERROR EXITS. PLEASE NOTE THAT THE
031500  % LOWER PART OF THE CORE ADDRESS REGISTER IS STORED IN DRIAR EVERY TIME THAT
031500  % REGISTER IS CHECKED, AND THIS IMPLIES THAT THE STATUS REGISTER MUST BE OK.
031500
031500  DRIAR, 0      % LOWER 16 BITS OF CORE ADDR REG AFTER TRANSFER (STATUS IS OK)
031501
031501  "8BDIM+8BDIS+8ZBDI
031501
031501  RSCON, 0      % SEEK CONDITION
031502
031502  "8BDIS+8BDIM+8ZBDI
```

```
031502
031502  % SPARE TRACK ALLOCATION HANDLER.
031502  % ACTIVATED WHEN A TRANSFER GIVES ADDRESS MISMATCH STATUS ERROR.
031502  % CHECKS BIT 4 IN TYPEC ,B TO SEE IF SPARE TRACK ALLOCATION IS ALLOWED.
031502  % THE SPARE TRACK ALLOCATION PARAMETERS (I.E. THE NEW CYLINDER AND SURFACE)
031502  % ARE READ FROM THE BAD TRACK, BY READING WITH THE SAME CYLINDER AND SURFACE
031502  % AS IN THE ORIGINAL ACCESS, BUT WITH SECTOR 0100, 0101, AND SO ON, UNTIL
031502  % TWO WORDS ARE READ CORRECTLY. THOSE TWO WORDS (CYLINDER AND SURFACE) ARE THEN
031502  % USED, TOGETHER WITH THE ORIGINAL SECTOR INCREMENTED BY 0100, TO ACCESS THE
031502  % SPARE TRACK.
031502
031502  SPHAN=*
031502
031502  "8BDIS+8BDIM+8ZBDI
031502
031502          LDA    SPAFL ,B          % THE SPARE TRACK ALLOCATION FLAG CAN BE 0/1/2
031503          JAF    L27
031504
031504  "8BDIS+8BDIM+8ZBDI
031504
031504          LDA    SSTAT             % ADDRESS MISMATCH ?
031505          BSKP   ONE 0100 DA
031506          JMP I (L33               % NO
031507  BRSC8,  INIOX RSC                % READ SEEK CONDITION
031510          BSKP   ZRO 0130 DA
031511          JMP    ERR               % SEEK ERROR
031512          SAA    1                 % FLAG:=1
031513          STA    SPAFL ,B
031514          LDA    CMAD1 ,B          % SAVE PART OF THE DATA FIELD
031515          STA    OCMD1 ,B
031516          LDA    CMAD2 ,B
031517          STA    OCMD2 ,B
031520          LDA    SVLBA ,B
031521          STA    OSVBA ,B
031522          LDA    SVLBB ,B
031523          STA    OSVBB ,B
031524          LDA    SVLCO ,B
031525          STA    OSVCO ,B
031526          LDA    SVLWC ,B
031527          STA    OSVWC ,B
031530          LDA    SVLWK ,B
031531          STA    OSVWK ,B
031532          LDA    SVLBA ,B          % SET SECTOR 0100
031533          AND    (177400
031534          BSET   ONE 060 DA
031535          STA    SVLBA ,B
031536          LDA    SVLCO ,B          % PUT READ-OP IN CONTROL WORD
031537          AND    (163777
031540          STA    SVLCO ,B
031541          AND    (177740           % AND IN BTSTA
031542          STA    BTSTA
031543  L25,    SAA    2
031544          STA    SVLWC ,B          % WORD COUNT := 2
031545          STZ    SVLWK ,B
031546          STZ    CMAD1 ,B          % MEMORY ADDR IN DATA FIELD
031547          COPY   SB DA
031550          AAA    NWLBB
031551          STA    CMAD2 ,B          % NWLBB AND NWLBA
031552          JMP I (BOM               % TRANSFER
031553  L27,    AAA    -1
```

```
031554           JAF    L31             % NONZERO: SPAFL IS NOT 1
031555           LDA    SSTAT           % STATUS ERROR ?
031556           AND    (017760
031557           JAF    L30             % NONZERO: YES. TRY NEXT SECTOR
031560           JPL I  (BRSR6          % NO. CHECK THE CORE ADDR REG
031561           JMP    L30             % CORE ADDR REG ERROR
031562           LDA    NWLBB ,B        % CHECK THE NEW CYLINDER NUMBER
031563           JAN    L30             % NEG: ILLEGAL
031564           SUB I  (BMXCY
031565           AAA    -1
031566           JAP    L30             % POS: ILLEGAL
031567           LDA    TRG ,B
031570           BSKP   ZRO 0160 DA
031571           JMP    L28             % PHOENIX
031572           LDA    NWLBA ,B        % CHECK THE NEW SURFACE
031573           JAN    L30             % NEG: ILLEGAL
031574           SHA    070
031575           AAA    1
031576           MPY I  (BSECT
031577           SUB I  (BSECY
031600           AAA    -1
031601           JAP    L30             % POS: ILLEGAL
031602           JMP    L29             % THE NEW CYL AND SURFACE OK (NOT PHOENIX)
031603
031603  )FILL
031615
031615  L28,     LDA    NWLBA ,B        % CHECK PHOENIX SURFACE
031616           JAN    L30             % NEG: ILLEGAL
031617           SHA    070
031620           JAZ    L29             % ZERO: OK
031621           AAA    -020
031622           JAN    L30             % NEG: ILLEGAL
031623           AAA    -5
031624           JAP    L30             % POS: ILLEGAL
031625  L29,     SAA    2               % FLAG:=2
031626           STA    SPAFL ,B
031627           LDA    OCMD1 ,B        % RESET DATA FIELD
031630           STA    CMAD1 ,B
031631           LDA    OCMD2 ,B
031632           STA    CMAD2 ,B
031633           LDA    OSVCO ,B
031634           STA    SVLCO ,B
031635           AND    (177740
031636           STA    BTSTA
031637           LDA    OSVWC ,B
031640           STA    SVLWC ,B
031641           LDA    OSVWK ,B
031642           STA    SVLWK ,B
031643           LDA    NWLBB ,B
031644           STA    SVLBB ,B        % NEW CYL
031645           LDA    NWLBA ,B
031646           AND    (177400
031647           STA    NWLBA ,B
031650           SAA    077
031651           AND    OSVBA ,B        % OLD SECTOR (+0100, OF COURSE)
031652           AAA    0100
031653           ADD    NWLBA ,B        % NEW SURFACE
031654           STA    SVLBA ,B
031655           JMP I  (BOM            % TRANSFER
031656  L30,     LDA    SVLBA ,B        % TRY NEXT SECTOR, IF POSSIBLE
```

```
031657          AAA    1
031660          STA    SVLBA ,B
031661          AND    (0377
031662          SUB I  (BSECT
031663          AAA    -0100
031664          JAN    L26              % NEG: TRY NEXT
031665          LDA    OCMD1 ,B         % RESET DATA FIELD
031666          STA    CMAD1 ,B
031667          LDA    OCMD2 ,B
031670          STA    CMAD2 ,B
031671          LDA    OSVBA ,B
031672          STA    SVLBA ,B
031673          LDA    OSVBB ,B
031674          STA    SVLBB ,B
031675          LDA    OSVCO ,B
031676          STA    SVLCO ,B
031677          LDA    OSVWC ,B
031700          STA    SVLWC ,B
031701          LDA    OSVWK ,B
031702          STA    SVLWK ,B
031703          JMP I  (ERR
031704  L31,    AAA    -1
031705          SKP    DA EQL 0
031706          JMP I  (ERR             % SPAFL IS NOT 2
031707          LDA I  (SSTAT
031710          AND    (017760
031711          JAF    L32              % NONZERO: STATUS ERROR
031712          JPL I  (BRSR6           % CHECK THE CORE ADDR REG
031713          JMP I  (ECADR           % ERROR
031714          LDA    OSVBA ,B         % RESET DISK ADDRS
031715          STA    SVLBA ,B
031716          LDA    OSVBB ,B
031717          STA    SVLBB ,B
031720          STZ    SPAFL ,B         % FLAG:=0
031721          MIN    SPACO ,B         % INCREMENT COUNTER
031722          0
031723  L32,    JMP I  (L33             % BACK TO DRIVER
031724
031724  )FILL
031737
031737  "8ZBDI -8BDIS
031737
031737  )PCL QDLAY
031737  )KILL RCA RSC LBA RSR LCO LWC LCA INIOX MORE9 NOIOX
031737
031737  "8ZBDI -8BDIS
031737
031737  % IN THE SWAP DRIVER, PLDWO AND PSTWO ARE NOT USED.
031737
031737  "8ZBDI -8BDIS
031737  )KILL 9ESWP; 9ESWP=*
031737  "
031737
031737
031737
031737  )LINE031737   @LIB OLDX
031737
031737
031737  "=============================================================================
031737  " 36.22       S T A R T
```

```
031737   %
031737   % RT-PROGRAM FOR STARTING SINTRAN
031737   %
031737
031737   INTEGER FPS41                              % FIRST PHYSICAL PAGE FOR SEGMENT 41
031740
031740   DOUBLE DSBANK=SBANK
031740   SUBR START
031740   INTEGER PARTIMER:=("TIMRT","1","2")
031743
031743   CRWTR:         FOR X DO
031743                    X=:D; MASSUNIT(0) SH 6\/DSKTYP+60=:XFUNC
031752                    T:=MASSNO(X); "RPAR"; *MON 2ABST
031755                    IF <0 THEN CALL ERRFATAL FI
031757                    XFUNC+1=:XFUNC; "WPAR"; *MON 2ABST
031764                    IF <0 THEN CALL ERRFATAL FI; X:=D
031767                    AD:=DTBLCK; T:=NBLCK; D+T; A:=A+C; AD=:DTBLCK
031774                    AD:=DFBLCK; T:=NBLCK; D+T; A:=A+C; AD=:DFBLCK
032001                  OD; EXIT
032003   *)FILL
032017
032017   INTEGER CDFADDR=?,CDBPCOLDSTART=?,CCHDEV=?
032017   START: "DTO1R"=:CDFADDR
032021          IF HENTFLAG=0 AND LGCOLDSTART><1 THEN
032027             CALL LOGPH
032030             IF A=0 OR A.TYPRING NBIT 5TERM THEN 1=:LGCOLDSTART; GO L1 FI
032040             X=:CDFADDR
032041             T:="DBPROG"; CALL XGTDFADDR; A=:CDBPCOLDSTART
032044             T:="DBPROG"; A:=0; CALL XSTDFADDR
032047             X:="DTO1R"; A:=0; T:="DBPROG"; CALL XSTDFADDR
032053             T:="HDEV"; X:=CDFADDR; CALL XGTDFADDR; A=:CCHDEV
032057             T:="CNTREG"; CALL XGTDFADDR; T:=CCHDEV+DCONT; *IOXT
032064             GO L1; *)FILL
032076          FI
032076   L1:    "PARTIMER"; *MON 2INTV
032100          "PARTIMER"; *MON 2RT
032102
032102   % MAKE STSIN A BACKGROUND PROGRAM
032102          *PIOF
032103          1=:BACKGROUND; RTREF.ACTPRI BONE 5BACKGR=:X.ACTPRI
032111
032111   % FIND BUFFER PAGE FOR MOVING OP.COM SEGMENT
032111          T:=CORMBANK; X:=BPAG1; *LDATX DPAGP
032114          IF A-"5BFPAGE"=0 THEN
032116             *LDXTX DPAGL; LDATX DPAGP
032120          ELSE
032121             *LDATX DPAGP
032122          FI; *PION
032123          0=:D; A:=:D
032125          AD SH 12=:DSBANK       % USE FIRST OR SECOND SWAP-PAGE
032127          ABLPAGE(0)=:NBLCK
032132          MASSNO(X); CALL LOGPH; A.TYPRING/\177774+3=:X.TYPRING
032141          IF X."TRNSF"="BDISK" THEN                  % READ FORMAT CALL TO DRIVER
032145             42=:XFUNC; T:=MASSNO(0); "RPAR"; *MON 2ABST
032153          FI; GO OVER; *)FILL
032172
032172   INTEGER CDFADDR,CDBPCOLDSTART,CCHDEV
032175   INTEGER POINTER TTIF:=TTIFIELD,PTTNO:=TTNO
032177
032177   OVER:
```

```
032177   % MOVE OP.COM SEGMENT
032177           IF HENTFLAG=0 THEN     % START AFTER ")HENT"
032201   % MOVE PIT3 SEGMENT FROM THE SAVE SEGMENT (17) TO SEGMENT 41
032201                   "SG17".MADR*NBLCK=:T; DDBLST; D+T; A:=A+C; AD=:DFBLCK
032211                   "SG41".MADR*NBLCK=:T; DDBLST; D+T; A:=A+C; AD=:DTBLCK
032221                   X.LOGADR SHZ -10-=:X
032225                   CALL FAR CRWTR
032226   % MOVE COMMAND SEGMENT FROM SAVE-AREA TO SEGMENT 3
032226                   "5OPSEG*5SEGSIZE+XSEGS"=:X=:L
032231                   "5FYOP"*NBLCK=:T; DDASA; D+T; A:=A+C; AD=:DFBLCK
032240                   X.MADR*NBLCK=:T; DDBLST; D+T; A:=A+C; AD=:DTBLCK
032247                   L.LOGADR SHZ -10-=:X
032254                   CALL FAR CRWTR
032255                   "SYSST"*NBLCK=:T; DDASA; D+T; A:=A+C; AD=:DFBLCK
032264                   "SG5".MADR*NBLCK=:T; DDBLST; D+T; A:=A+C; AD=:DTBLCK
032274                   "SG5".LOGADR SHZ -10 -=:X
032301                   CALL FAR CRWTR
032302                   IF LGCOLDFLAG><1 THEN
032306                      A=:PTTNO; CDFADDR=:TTIF; CDBPCOLDSTART=:DBPCOLDSTART
032313                   FI
032313           FI
032313   @LIB CXCPU
032313           T:=CDFADDR.TDFPHPAGE
032315           RTREF.WINDOW/\177400\/T=:X.WINDOW
032322           CDFADDR.TDFLGADDR/\1777+"5UBFPAGE*2000"=:TTIF
032327           T=:"RTERR".WINDOW
032331   @ELIB
032331           GO OVFILL: *)FILL
032356
032356   OVFILL:
032356   % COPY PIT3 SEGMENT INTO MEMORY
032356           "SG41".MADR*NBLCK=:T; DDBLST; D+T; A:=A+C; AD=:DFBLCK
032366           X.LOGADR SHZ -10*NBLCK=:NBLCK
032372           MASSUNIT(0) SH 6\/DSKTYP+60=:XFUNC
032400           A:=FPS41=:D:=0; AD SH 12=:DSBANK
032405           T:=MASSNO(X); "RPAR"; *MON 2ABST
032410           IF A<0 THEN CALL ERRFATAL FI
032412           ABLPAGE(0)=:NBLCK
032415   % SET UP PAGE LINK FOR PIT3 SEGMENT
032415           "SG41".LOGADR SHZ -10=:L
032421           FPS41 SH 2+CORMSTART=:X.BPAGLINK
032425           X.LOGADR/\377=:D; X:=X.BPAGLINK; T:=CORMBANK
032432           DO WHILE L><0
032434              A:=X+4; *STDTX DPAGL
032437              A:=162041; *STATX DPGPR
032441              L-1; D+1; X+4
032444           OD; X-4; *STZTX
032447           "SG41".FLAG BONE 5FIX=:X.FLAG
032453   % SET UP PAGE TABLE FOR PIT3 SEGMENT
032453           T:=CORMBANK; X.LOGADR/\77 SH 1+177600=:B; X:=X.BPAGLINK; CALL SS41IPIT
032463           GO IOBHDLC; *)FILL
032507
032507   INTEGER CLOGU,NPAGES,CPNTR,CFPAG,CLPG
032514   INTEGER PM061:=("5",NPAGES,CFPAG,CLPG,0,0)
032522   INTEGER POINTER LREG
032523
032523   SUBR: A:=L=:"LREG"; X:=0
032526           DO WHILE X<<"CUMSIZE*2"
032531              X=:L
032532              A:="CUMTABLE"; X+A; T:=0; *LDDTX
```

```
032536              IF A<<=CFPAG AND D+1>>T THEN T:=D=:CFPAG FI
032546              IF X:=CFPAG>>=CLPG GO LREG
032552              IF A<<=CLPG AND A>>=CFPAG THEN A-1=:CLPG FI
032562              X:=L+2
032564           OD; MIN "LREG"; GO LREG
032567
032567    IOBHDLC: "IOBUTAB"=:CPNTR
032571           DO WHILE CPNTR.S0><-1
032576              IF A BIT BIHDLC AND X.S1/\77777>>1777 THEN
032605                 A+1777 SHZ -12=:NPAGES
032610                 CPNTR.S0/\37777; CALL LOGPH
032614                 IF T:=X.S0 BIT 17 THEN A:=D FI; IF A=0 THEN CALL ERRFATAL FI
032622                 A=:CLOGU; -1=:CLPG
032625    DOLOOP:      DO CLPG+1=:CFPAG WHILE CFPAG<<ENDPAGE
032634                    CFPAG+77=:CLPG; CALL CSUBR; GO DOLOOP; "PM061"; *MON 61
032643                    GO DOLOOP; GO FOUND
032645                 OD; CALL ERRFATAL
032647    FOUND:       T=:CLOGU.FIXID; A=:X.LWPHY=:D:=0; AD SH 12
032655                 A=:X.MASTB:=D=:X.BUFST; 1=:X.SWBUF; 0=:X.HINIF
032663                 NPAGES SH 13=:X.MAX
032666              FI; CPNTR+2=:CPNTR
032671           OD
032672          GO OLDSTART        % ON OP. SEGMENT
032673    RBUS
032706
032706    %=============================================================================
032706    %  38.10      S Y S E V A L
032706    %
032706    %  ROUTINE TO DEDUCT CPU/SYSTEM INFORMATION AND UPDATE THE GLOBAL
032706    %  SINTRAN VARIABLES WITH THIS INFORMATION
032706    %
032706    %  A 12 WORD (14 OCTAL) ARRAY CONTAIN VARIOUS INFORMATION ABOUT THE SYSTEM.
032706    %  THE ARRAY IS SUBDIVIDED INTO SEVERAL FIELDS FOR VARIOUS TYPE OF INFORMATION.
032706    %  THE FOLLOWING GIVES A DETAILED DESCRIPTION OF THE INFORMATION PRESENT IN
032706    %  THE VARIOUS ENTRIES
032706    %
032706    %-----------------------------------------------------------------------------
032706    %
032706    %  DISP   NAME              DESCRIPTION
032706    %
032706    %   0     SYSNO             SYSTEM NUMBER (NORMALLY CPU NUMBER), 16 BIT INTEGER
032706    %
032706    %   1     HWINFI(0)         HARWARE INFORMATION
032706    %                           LEFT BYTE = CPU TYPE
032706    %                                       0 = NORD-10  48 BIT FLOATING
032706    %                                       1 = NORD-10  32 BIT FLOATING
032706    %                                       2 = ND-100   48 BIT FLOATING
032706    %                                       3 = ND-100   32 BIT FLOATING
032706    %                                       4 - 255  NOT USED
032706    %                           RIGHT BYTE = INSTRUCTION SET
032706    %                                       0 = STANDARD (NORD-10 OR ND-100)
032706    %                                       1 = NORD-10 COMMERCIAL, ND-100/CE
032706    %                                       2 - ND-100/CE WITH "MICRO-SEGADM"
032706    %                                       3 - 255 NOT USED
032706    %
032706    %   2     HWINFO(1)         NOT USED
032706    %
032706    %   3     HWINFO(2)         NOT USED
032706    %
032706    %   4     SINVER(0)         OPERATING SYSTEM
```

```
032706   %                          LEFT BYTE = OPERATING SYSTEM
032706   %                                      0 = SINTRAN III VS
032706   %                                      1 = SINTRAN III VSE
032706   %                                      2 = SINTRAN III VSE/500
032706   %                                      3 = SINTRAN III RTP
032706   %                                      4 = SINTRAN III VSX
032706   %                                      5 = SINTRAN III VSX/500
032706   %                                      6 - 255 NOT USED
032706   %                          RIGHT BYTE = VERSION LETTER
032706   %                                       ASCII CHARACTER WITHOUT PARITY (A-Z)
032706   %
032706   %    5     SINVER(1)       NOT USED
032706   %
032706   %    6     REVLEV          PATCH/CORRECTION LEVEL INDICATIOR, 16 BIT INTEGER
032706   %                          (SYSTEM DEPENDANT CODING)
032706   %
032706   %    7     GENDAT(0)       SYSTEM GENERATION TIME  (MINUTES)
032706   %    8     GENDAT(1)       SYSTEM GENERATION TIME  (HOURS)
032706   %    9     GENDAT(2)       SYSTEM GENERATION TIME  (DAY)
032706   %   10     GENDAT(3)       SYSTEM GENERATION TIME  (MONTH)
032706   %   11     GENDAT(4)       SYSTEM GENERATION TIME  (YEAR)
032706   %
032706   %-------------------------------------------------------------------------
032706
032706   SUBR SYSEVAL
032706
032706   INTEGER SXREG,SBREG,STREG,AREG,SDREG
032713   TRIPLE TADR=STREG
032713   INTEGER POINTER SLREG
032714
032714   SYSEVAL: TAD=:TADR; X=:SXREG:=B=:SBREG:=L=:"SLREG"
032722   %  32/48 BIT FLOATING?
032722            T:=0; A:=1; *NLZ 20
032725            IF T=0 THEN T:=1 ELSE T:=0 FI
032732
032732   %  NORD-10 OR ND-100?
032732            A:=CPSTA/\10000 SHZ -13+T SH 10=:HWINFO(0)
032741
032741   %  INSTRUCTION SET
032741   %  COMMERCIAL INSTRUCTION SET?
032741            *TRA IIC
032742            A:=20; *TRR IIE
032744            A:=0; *TRR PIE; TRR PID
032747            A:=40000; *MST PIE
032751            A:="CLEV14"; *IRW 160 DP; ION
032754            T:="L1"=:L:=0; *140130; JMP *+1; TRA IIC              % BFILL
032762   L1:      IF A=0 THEN
032763               "L2"=:L
032765               B:=0; D:=0; A:=0; X:=0; *142700; TRA IIC          % GECO
032773   L2:         IF A><0 GO OUT
032774            FI
032774            *143500                                              % SLWCS
032775            MIN HWINFO(0)
033000            T:="L3"=:L; X:=0; *ICLEP; TRA IIC
033004   L3:      IF A=0 THEN MIN HWINFO(0) FI
033007   OUT:
033007   *'-CXCPU
033007            A:=4
033010   *"
033010            IF T:=PN500D><0 THEN A+1 FI                          % ND-500 INCLUDED
```

```
033014          A SH 10+##J                                          % VERSION LETTER
033016          A=:SINVER(0)
033020          *IOF; TRA IIC
033022          A:=0; *TRR PIE; TRR PID; TRR IIC
033026          TAD:=IADR; X:=SBREG=:B:=SXREG; GO SLREG
033033
033033  % ROUTINE ON LEVEL 14 TO CHECK FOR ILLEGAL INSTRUCTION
033033  CLEV14: *TRA IIC
033034          IF A><4 THEN CALL ERRFATAL FI
033040          *IRW DA; IRR DL; IRW DP
033043          *WAIT
033044          GO CLEV14
033045
033045  RBUS
033060  @LIB OLDX
033060  %=============================================================================
033060  %       I N T D F I E L D S
033060  %
033060  % INITIALIZE THE TERMINAL DATAFIELDS
033060  %
033060
033060  INTEGER 77CBU                           % END OF "POF-RTDESC"+1
033061
033061  SUBR INTDFIELDS
033061
033061  INTEGER ARRAY DAMO(0)
033061          *37
033062  INTEGER FEB
033063          *37;36;37;36;37;37;36;37;36;37
033075
033075  @LIB CXCPU
033075  INTEGER POINTER P1CDFADDR;=1CDFADDR
033076  DOUBLE POINTER PDCDFADDR=P1CDFADDR
033076  INTEGER CINDX=?,1CDFADDR=?,2CDFADDR=?
033076  DOUBLE DCDFADDR=?
033076  @ELIB
033076
033076  % COMPUTE THE NUMBER OF DAYS IN FEBRUARY THE YEAR THE SYSTEM
033076  % IS GENERATED
033076  INTDFIELDS:
033076          GENDA(4)-PTBASE(X); D:=0
033102          DO WHILE A>0
033104             A-1; IF D+1>3 THEN D:=0 FI
033112          OD; IF D=1 THEN 35 ELSE 34 FI; A=:FEB
033122          X:=GENDA(3)-1; A:=DAMO(X)+1-=:MND
033131          *PIOF
033132          CALL CH5CPUPRESENT                  % CHECK IF ND-500 CPU IS PRESENT
033133          A:=200; *TRR IIE; TRR IIC
033136          A:=10; *TRR ECCR; TRA IIC
033141          IF A><0 THEN
033142             0=:TCNTI
033143          ELSE
033144             A:=4; *TRR ECCR
033146          FI; A:=0; *TRR IIE; TRA PGS; TRA PEA; TRA IIC
033153  * BBACS
033153           A:="THISS"-"BAK01"=:D:=0; T:=5RTSIZE; *RDIV ST
033161          IF D><0 THEN CALL ERRFATAL FI; A=:1NBPP
033165          A:="2THSS"-"BAK01"=:D:=0; T:=5RTSIZE; *RDIV ST
033173          IF D><0 THEN CALL ERRFATAL FI; A=:2NBPP
033177  *
```

```
"033177          GO L0; *)FILL
033215
033215    @LIB CXCPU
033215    INTEGER ARRAY POINTER PDFTBIT=?
033215    INTEGER TRLND=?,BINSZ=?,BOUSZ=?,PGPART=?
033215    TRIPLE TIOBE=?; DOUBLE TBUSZ=?
033215    DISP 0; TRIPLE DTIOBE; PSID
033215    @ELIB
033215
033215    L0:
033215
033215    @LIB CXCPU
033215          A:=-1; FOR X:=0 TO 17 DO A=:PDFTBIT(X) OD
033225          A:=TDLPHPAGE-TDFPAGE+1*4=:D:=0; T:=20; *RDIV ST
033235          X:=0; DO WHILE X><A; 0=:PDFTBIT(X); X+1 OD
033243          A:=-1; T:=174005
033245          DO WHILE D><0
033247             *EXR ST
033250             D-1; T+10
033252          OD; A=:PDFTBIT(X)
033254          4=:PGPART; GO STLOP; *)FILL
033263
033263    INTEGER ARRAY POINTER PDFTBIT:=DFTBIT
033264    INTEGER POINTER PCINDX:=CINDX, PTRLND:=TRLND
033266    TRIPLE POINTER PTIOBE:=TRLND; DOUBLE POINTER PTBUSZ:=TBUSZ
033270    INTEGER TIOBADR,PGPART
033272    INTEGER DFTINDX;=0
033273    INTEGER DFTELM,DFTEND
033275    INTEGER FREBIT,BTNUM,SAVX
033300
033300    STLOP: "TIOBU"=:TIOBADR
033302    LOOP:  TIOBADR.DTIOBE=:PTIOBE
033305           IF A=-1 THEN
033310              PGPART -1=:PGPART; IF A=0 THEN GO FAR RETU ELSE GO STLOP FI
033316           FI
033316           TIOBADR+3=:TIOBADR
033321           PTBUSZ; A+D+TDISIZ+TDOSIZ; T:=0; DO A-400; T+1; WHILE A>0 OD
033333           IF T>A:=4 THEN CALL ERRFATAL FI
033337           IF PGPART><T GO LOOP
033342           PTRLND; CALL LOGPH; IF A=0 THEN CALL ERRFATAL ELSE A=:PCINDX FI
033350           DFTINDX=:DFTEND; PGPART=:L
033354    DLOOP: PDFTBIT(DFTINDX)=:DFTELM
033357           IF A><-1 THEN
033362              T:=175005; 0=:FREBIT=:BTNUM
033365              FOR X:=-20 DO
033366                 A:=DFTELEM; *EXR ST
033370                 GO NTFBIT; MIN FREBIT
033372                 IF FREBIT=L THEN
033375                    X=:SAVX:=4; A:=BTNUM=:D:=0; *RDIV ST
033403                    IF D+1>=L THEN
033406                       A=:X:=DFTINDX*4+X=:X; A:=D-L*400; A:=:X; A*2000+X=:L
033422                       BTNUM SHZ 3; T:=174205; T+A; DFTELM
033427                       FOR X:=FREBIT- DO; *EXR ST
033432                         T-10
033433                       OD
033434                       A=:PDFTBIT(DFTINDX)
033436                       GO FOUND
033437                    FI
033437                    A:=D=:FREBIT; X:=SAVX
033442                 FI; GO FBIT
```

```
033443   NTFBIT:       0=:FREBIT
033444   FBIT:         T+10; MIN BTNUM
033446              OD
033447           FI
033447           IF DFTINDX+1=20 THEN "0" FI; A=:DFTINDX
033456           IF A=DFTEND THEN CALL ERRFATAL FI
033462           GO DLOOP; *)FILL
033474
033474   INTEGER CINDX,1CDFADDR,2CDFADDR; DOUBLE DCDFADDR=1CDFADDR
033477   INTEGER TRLND,BINSZ,BOUSZ; TRIPLE TIOBE=TRLND; DOUBLE TBUSZ=BINSZ
033502
033502   FOUND: A:=L; X:=CINDX; CALL MAKETDFS(TRLND); GO FAR LOOP
033507
033507   INTEGER ARRAY DFTBIT(20)
033527   *)FILL
033531
033531   @ELIB
033531   RETU:  7RTDLGADDR=:D
033533           FOR X:=RTSTART STEP 5RTSIZE TO SEGSTART-5RTSIZE DO
033540              A:=D=:X.RTDLGADDR+5XRTDSIZE=:D
033544           OD
033546           FOR X:=7RTDLGADDR TO 77CBUF-1 DO 0=:X.S0 OD
033556           "DUMMY".STADR=:X.RTDLGADDR.DPREG
033562           "STSIN".STADR=:X.RTDLGADDR.DPREG
033566           "1SWAP".STADR=:X.RTDLGADDR.DPREG
033572           GO SETPTABL
033573   RBUS
033603
033603   @LIB OLDX
033603
033603   @EOF
033603
```

```
033603   %
033603   %=============================================================================
033603   %      P I T 3 - S E G M E N T
033603   %=============================================================================
033603
033603
033603   *9EPT3+1777@-12@12/
114000   *XSPT3=*
114000   *XSPT3;XLPT3;XEPT3
114003   *S41FP=XSPT3@-12+100
114003
114003
114003   %=============================================================================
114003   %             P F I X C
114003   %
114003   % RESERVE MEMORY FOR SPECIAL USE
114003   %
114003   SUBR PFIXC
114003
114003   DISP -200
114003   INTEGER SGADR,FPHYS,SG1ADR,OLDFLAG
114003   INTEGER CMMENTRY,CSEGN,STEPINCR,CCORMSTART,CECORMAP,BREG
114003   INTEGER COLDFLAG,CINDX
114003   INTEGER PGFOUND,PREVPGFOUND
114003   INTEGER CFPHYS,CT,CA,CD,CLOGPAGE,CCECORMAP
114003   INTEGER POINTER LINK
114003   TRIPLE CTAD=CT; DOUBLE CTA=CT
114003   INTEGER FIXFLAG
114003   INTEGER NPAGES; INTEGER CCTINDX=NPAGES
114003   INTEGER AFPHYS,NPARTS=AFPHYS
114003   INTEGER ALPHYS,CPHPAGE=ALPHYS; DOUBLE ARRAY POINTER DPARTADDR=ALPHYS
114003   INTEGER NSHARED; INTEGER POINTER RNPARTS=NSHARED; INTEGER MXMININTERVAL=NSHARED
114003   INTEGER ARRAY POINTER ARRPNT; DOUBLE ARRAY POINTER DRETADDR=ARRPNT
114003   DOUBLE ARRAY POINTER DARRPNT=ARRPNT
114003   TRIPLE ARRAY POINTER FARRPNT=ARRPNT
114003   INTEGER ARRAY POINTER PCCTAB
114003   DOUBLE ARRAY POINTER DPCCTAB=PCCTAB
114003   DOUBLE POINTER DCCFPA
114003   INTEGER PREVT,PREVX
114003   TRIPLE T5D1=NPAGES; DOUBLE D5DD4=NSHARED
114003   INTEGER CURPAGE,CPEND; DOUBLE CCAREA=CURPAGE
114003   INTEGER CNPAGES=CPEND, CSGNO=CPEND, CTINDX=CPEND
114003   INTEGER CBPGLINK
114003   INTEGER PARTINDX,RETINDX,CCPRG=PARTINDX
114003   INTEGER POINTER RINDXADDR
114003   INTEGER SVFLAGB
114003   PSID
114003
114003   INTEGER CBRG
114004
114004   PFIXC: A:=BREG:=:B=:CBRG
114007          CALL XTOPOFF(SRESER)                    % RESERVE "SWAPPING SYSTEM"
114011          0=:CINDX=:PAGPN; CBRG=:B
114015          CALL XTOPOFF(XCSEGS)
114017          RTREF; X:="FIXCRT"; T:=0; *STATX        % CURRENT PROGRAM EXECUTING FIXC
114023   @FICR
114023          FIXFLAG GOSW 0F500,FAR 1F500,FAR 2F500,FAR 3F500,FAR 4F500,
114032          FAR 5F500,FAR 6F500,FAR 7F500,FAR 10F500;
114036   @CR;
114036   0F500: CALL MBSREMOVE                  % REMOVE SEGMENT FROM MEMORY
```

```
114037          NPAGES-1+FPHYS=:CURPAGE
114043          ECORMAP=:CECORMAP
114045          CALL SGFIXC                         % FIX SEGMENT CONTIGUOUSLY
114046          FIXPAGES+NPAGES=:FIXPAGES
114051          CALL XRELES
114052          BREG=:B; IF DO BIT 17 THEN 0=:ZAREG FI; GO RETSTUPR
114061      *)FILL
114103
114103      INTEGER CB,XINDX
114105      INTEGER POINTER LREG
114106
114106      % RELEASE ALL SWAPPING-DEVICES AND THE SWAPPING-SYSTEM
114106      XRELES: A:=L=:"LREG"
114110          CALL XTOPOFF(SRELES)
114112          X:="FIXCRT"; T:=0; *STZTX
114115          IF BACKGROUND><0 AND SVFLAGB BIT 5ESCON THEN CALL ESCON FI
114123          GO LREG
114124
114124      % REMOVE ALL PAGES OF A SEGMENT FROM MEMORY
114124      MBSREMOVE: A:=L=:"LREG"
114126          IF SGADR.SEGLINK><0 THEN CALL XTOPOFF(SREMOVE) FI
114133          X.FLAG=:OLDFLAG BZERO 50K=:X.FLAG
114137      % REMOVE ALL PAGES CURRENTLY IN SEGMENT
114137          DO WHILE X:=SGADR.BPAGLINK><0
114142              T:=CORMBANK; *LDATX DPGPR
114144              IF A BIT 5WIP THEN A:=1=:D; T:=SGADR; CALL XTOPOFF(ATRNSEG) FI
114153              T:=SG1ADR; SGADR; CALL XLINKOVER
114156          OD; GO LREG
114160      *)FILL
114166
114166      %       ALLOCATE MEMORY FOR NORD-500
114166      IF500:
114166      500FIXC: CORMSTART=:CCORMSTART; AFPHYS=:CURPAGE
114172      %       FIND POSSIBLE AREA FOR NORD-500 SEGMENT
114172      XLOOP: X:=CCORMSTART; CURPAGE=:FPHYS; 0=:PGFOUND
114176      % SEARCH ALL PAGES IN MEMORY-MAP
114176          DO
114176              T:=CORMBANK; *LDDTX DPGPR
114200              X=:CMMENTRY
114201              IF D=CURPAGE THEN
114204                  IF A/\377=0 OR A>>SGMAX GO NEWSTART
114211                  A=:CSEGN*5SEGSIZE+SEGSTART=:SGADR
114215                  IF A.FLAG BIT 5FIX THEN
114221                      CALL TSTSHARED; GO 5ERR; GO NEWSTART
114224                  FI; GO PGOK
114225              ELSE
114226                  IF D>>T GO CHRTC
114230              FI
114230      INLOOP: WHILE X:=CMMENTRY<<ECORMAP
114234              X+4
114235          OD
114236      CHRTC: X:=0=:CSEGN
114240      % CHECK RT-COMMON
114240          DO AD:=DPCCTAB(X) WHILE A><-1
114244              IF D=CURPAGE THEN
114247                  A/\377=:T; AD:=DCCFPA
114252                  IF A-100=0 GO NEWSTART; A+200; D BONE 6
114256                  IF T<A OR T>D GO NEWSTART
114262                  CALL TSTSHARED; GO 5ERR; GO NEWSTART; CMMENTRY-4=:CMMENTRY; GO PGOK
114271              FI; X+2
```

```
114272          OD; GO NEWSTART
114274
114274    5ERR:  A=:BREG.ZAREG; CALL XRELES; BREG=:B; GO RETSTUPR
114302    *)FILL
114315
114315    % CURRENT AREA NOT AVAILABLE, START AFTER LAST CHECKED PAGE IF POSSIBLE
114315    NEWSTART: MIN CURPAGE
114316          IF CURPAGE-1+NPAGES>>ALPHYS THEN   % NO AREA AVAILABLE FOR NORD-500
114324             A:=0; GO 5ERR FI
114326          CMMENTRY=:CCORMSTART; GO XLOOP
114331
114331    % THIS PAGE IS AVAILABLE
114331    PGOK:  MIN PGFOUND; MIN CURPAGE
114333          IF PGFOUND><NPAGES GO INLOOP   % ALL NEEDED PAGES NOT YET FOUND
114337    % AREA FOR NORD-500 SEGMENT IS FOUND
114337          IF CMMENTRY><ECORMAP THEN A+4 FI; A=:CCECORMAP
114345          FPHYS=:AFPHYS+NPAGES-1=:ALPHYS  % PHYSICAL ADDRESSES OF NORD-500 AREA
114352          IF FIXFLAG=4 THEN
114356             AFPHYS=:FPHYS+NPAGES-1=:CURPAGE
114363             CSGNO*5SEGSIZE+SEGSTART=:SGADR
114367             CALL 5SGFIXC; FIXPAGES+NPAGES=:FIXPAGES
114373             AFPHYS; GO FAR OKRET; *)FILL
114403          ELSE
114404             IF A=5 THEN
114407                AFPHYS=:FPHYS+NPAGES-1=:CURPAGE
114414                CTINDX+1=:SGADR; CALL 5SGFIXC
114420                CTINDX=:BREG.ZTREG; X:=CTINDX+1; RTREF=:FXCTAB(X)
114427                AFPHYS; GO FAR OKRET
114431             FI
114431          FI; 0=:CINDX
114432    %  TEST IF THE SHARED AREAS ARE OF CORRECT SIZE AND IF THEY ARE PLACED
114432    %  IN CORRECT PHYSICAL ADDRESSES
114432    LOOP2: IF CINDX=NSHARED GO FAR FINEX   % FOR CINDX DO WHILE CINDX><NSHARED
114436             CINDX*3=:X                    % ; TAD:=FARRPNT(X)=:CTAD
114441             T:=0; A:="FARRPNT"; X+A; *LDDTX
114445             AD=:CTA; *LDATX 20
114447             A=:CD
114450             IF SGMAX<<CT THEN 311; GO 5ERR FI      % ILL. SEGM. NO.
114456             IF T=0 THEN                   % RT-COMMON
114460                IF CCFPA=100 THEN 302; GO 5ERR FI   % NO RT-COMMON
114466                A+100=:CLOGPAGE
114470                IF CCLPA-CCFPA+1><CD THEN 312; GO FAR 5ERR FI % RT-C SIZE DOES NOT MATCH
114500                AFPHYS+CA=:CURPAGE; 0=:PGFOUND
114504    LOOP1:      X:=0; PGFOUND=:PREVPGFOUND
114507                DO AD:=DPCCTAB(X) WHILE A><-1
114513                   IF A/\377=CLOGPAGE THEN
114517                      IF D><CURPAGE THEN 303; GO FAR 5ERR FI  % RT-C IN ILL. PHYS ADDRESS
114524                      MIN PGFOUND
114525                      IF CLOGPAGE=CCLPA+100 THEN
114532                         IF PGFOUND><CD THEN 314; GO FAR 5ERR FI % RT-C NOT CONTIGOUS
114540                         GO FAR NEXT
114541                      FI; MIN CURPAGE; MIN CLOGPAGE
114543                   FI; X+2
114544                OD; IF PGFOUND><PREVPGFOUND GO LOOP1
114551                304; GO FAR 5ERR; *)FILL      % ERROR IN LINKING TO RT-COMMON
114575             ELSE
114576                A:=T*5SEGSIZE+SEGSTART=:X
114602                IF X.FLAG BIT 5DEMAND THEN A:=T SH 10+313; GO FAR 5ERR FI
114611                IF X.LOGADR SHZ -10><CD THEN CT SH 10+316; GO FAR 5ERR FI
114622                IF X.FLAG BIT 5FIX THEN
```

```
114625                    AFPHYS+CA=:CURPAGE
114630                    IF X:=X.BPAGLINK=0 THEN CT SH 10+305; GO FAR 5ERR FI
114637                    DO
114637                       T:=CORMBANK; *LDATX DPAGP
114641                       IF A><CURPAGE THEN CT SH 10+307; GO FAR 5ERR FI
114650                       MIN CURPAGE; T:=CORMBANK; *LDXTX DPAGL
114653                    WHILE X><0
114654                    OD
114655                 FI
114655              FI
114655    NEXT:  CINDX+1=:CINDX; GO FAR LOOP2
114661    FINEX: 0=:CINDX
114662    % MOVE ALL SHARED SINTRAN III SEGMENTS, NOT YET FIXED, FROM MEMORY
114662           FOR CINDX DO WHILE CINDX><NSHARED
114666              A*3=:X
114670              "ARRPNT"; X+A; T:=0; *LDATX             % IF ARRPNT(X)><0 THEN
114674              IF A><0 THEN
114675                 A*5SEGSIZE+SEGSTART=:SGADR; CALL MBSREMOVE
114701              FI
114701           OD
114703           0=:CINDX=:CLOGPAGE
114705           IF NSHARED><0 THEN
114707              A*3=:X; "ARRPNT"; X+A; T:=0; *STZTX     % 0=:ARRPNT(X) FI
114715           FI; GO RM500; *)FILL
114731    % RESERVE THE MEMORY AREA FOR NORD-500 SEGMENT
114731    RM500: FOR CINDX DO WHILE CINDX><NSHARED
114735              CINDX*3=:X; T:=0; "FARRPNT"; X+A; *LDDTX    % TAD:=FARRPNT(X)=:CTAD
114744              AD=:CTA; *LDATX 20
114746              A=:CD; TAD:=CTAD
114750              IF A><CLOGPAGE THEN
114753    % NOT SHARED AREA, LINK TO NORD-500 PAGELINK
114753                 A-T=:CNPAGES-1+CLOGPAGE+AFPHYS=:CURPAGE
114761                 CLOGPAGE+AFPHYS=:FPHYS; 0=:SGADR
114765                 CALL 5SGFIXC; CLOGPAGE+CNPAGES=:CLOGPAGE
114771              FI
114771              IF CT=0 THEN                    % RT-COMMON
114773                 CLOGPAGE+CD=:CLOGPAGE
114776              ELSE
114777                 A*5SEGSIZE+SEGSTART=:SGADR=:X
115003                 IF X.FLAG NBIT 5FIX THEN  % SHARED SEGMENT NOT YET FIXED
115006                    CA+AFPHYS=:FPHYS+CD-1=:CURPAGE
115014                    CALL 5SGFIXC              % FIX SEGMENT
115015                    CD+FIXPAGES=:FIXPAGES
115020                    NSHARED*3=:X=:D; T:=0; "ARRPNT"; X+A        % MIN ARRPNT(X); ARRPNT(X); X+A
115027                    *LDATX; AAA 1; STATX
115032                    X:=A+D                                      % =ARRPNT(X)
115034                    "ARRPNT"; X+A; CT; *STATX                   % CT=:ARRPNT(X)
115040                 FI; CLOGPAGE+CD=:CLOGPAGE
115043              FI
115043           OD
115045           IF NPAGES-CLOGPAGE><0 THEN
115050    % NOT SHARED AREA, LINK TO NORD-500 PAGLINK
115050              A=:NPAGES; CLOGPAGE+AFPHYS=:FPHYS+NPAGES-1=:CURPAGE
115057              0=:SGADR; CALL 5SGFIXC; PAGPN=:BREG.ZTREG; 0=:PAGPN
115065           FI; AFPHYS
115066    OKRET: A=:BREG.ZAREG; MIN X.ZPREG; 0/\0; CALL FAR XRELES; BREG=:B
115075           GO RETSTUPR
115076
115076    *)FILL
115106
```

```
115106
115106   % FIND WANTED PAGES IN MEMORY-MAP, REMOVE THEM FROM THEIR CURRENT SEGMENT
115106   % AND INSERT PAGES INTO FIXC-SEGMENT
115106   SSGFIXC: CCECORMAP=:CECORMAP
115110   SGFIXC: A:=L=:"LINK"
115112           4=:STEPINCR; CORMSTART=:CCORMSTART
115116   LOOP:   X:=CCORMSTART
115117           DO
115117              T:=CORMBANK; *LDDTX DPGPR
115121              IF D=CURPAGE THEN
115124                 IF A/\377=0 GO ERR                    % PAGE NOT IN SWAPPING AREA
115126                 A*5SEGSIZE+SEGSTART=:CSEGN; X=:CMMENTRY
115132                 IF A.FLAG BIT 5FIX GO ERR             % PAGE BELONGS TO FIXED SEGM.
115136                 X:=CMMENTRY; CSEGN; CALL OUTLINK
115141                 T:=CORMBANK; *LDATX DPGPR
115143                 IF A BIT 5WIP THEN
115145                    CSEGN.FLAG=:COLDFLAG BZERO 5OK=:X.FLAG; X:=CMMENTRY
115153                    T:=CSEGN; A:=1=:D; CALL XTOPOFF(ATRNSEG)
115160                    IF CSEGN.FLAG BIT 5DEMAND THEN COLDFLAG=:X.FLAG FI
115166                 FI; GO FORBI; *)FILL
115176   FORBI:         X:=CMMENTRY; T:=SGADR; CSEGN; CALL XLINKOVER
115202                 IF CURPAGE=FPHYS GO L1
115206                 A-1=:CURPAGE
115210                 IF STEPINCR>0 THEN
115213                    -4=:STEPINCR; CMMENTRY-4=:CCORMSTART
115220                    T:=NPAGES-2 SH 2; A-T; A=:CECORMAP
115225                 FI; GO LOOP
115226              FI
115226           WHILE X><CECORMAP
115231              A:=X+STEPINCR=:X
115234           OD
115235   % AREA NOT AVAILABLE
115235   ERR:    IF FIXFLAG><0 THEN -1; GO FAR 5ERR FI
115241           IF X:=SGADR.BPAGLINK><0 THEN
115244              T:=CORMBANK; A:=1
115246              DO WHILE X><0
115247                 X=:D; *STATX DPGPR; STZTX DALOG; LDXTX DPAGL
115253              OD; "XSGRT".BPAGLINK; X:=D; *STATX DPAGL
115260              SGADR.BPAGLINK; D=:X.BPAGLINK; A=:"XSGRT".BPAGLINK
115265           FI
115265           CALL XRELES; BREG=:B
115270           IF DO BIT 17 THEN A:=-1; A=:ZAREG; GO RETSTUPR FI
115276           CALL 9ERRA(#28); GO RETXIT
115301
115301   L1:     IF X:=SGADR>>SEGSTART THEN
115305              X.FLAG BONE 5FIX BONE 5OK=:X.FLAG; X.LOGADR/\377=:CURPAGE
115314              A:=X-SEGSTART; A=:D:=0; T:=5SEGSIZE; *RDIV ST
115322              A=:D; X.FLAG/\177400\/D=:D; X:=X.BPAGLINK
115330              T:=CORMBANK; A:=CURPAGE
115332   % SET CORRECT LOGICAL PAGE NUMBERS AND SEGMENT NUMBER IN CORMAP ENTRIES
115332              DO
115332                 *STDTX DALOG
115333                 A+1; *LDXTX DPAGL
115335              WHILE X><0
115336              OD
115337   % TRANSFER SEGMENT TO MEMORY
115337              A:=SGADR.LOGADR SHZ -10
115342              X:=X.BPAGLINK; D:=0; T:=SGADR; CALL XTOPOFF(ATRNSEG)
115347           FI; GO LINK; *)FILL
115364
```

```
115364    ! TEST IF THE SHARED SINTRAN III/NORD-500 AREA ARE PLACED IN CORRECT
115364    ! PLACE IN PHYSICAL MEMORY, AND TEST IF THE AREA SIZES ARE CORRECT
115364    TSTSHARED: 0=:CINDX
115365           FOR CINDX DO WHILE CINDX><NSHARED
115371              CINDX*3=:X; "FARRPNT"; X+A; T:=0; *LDATX    % TAD:=FARRPNT(X)=:CTAD
115400              AD=:CTA; *LDATX 20
115402              A=:CD; TAD:=CTAD
115404              IF CSEGN=T THEN
115407                 IF A=0 THEN                    % RT-COMMON
115410                    IF CCFPA=100 THEN EXIT FI          % ERROR, NO RT-COMMON
115415                    A+100=:CLOGPAGE; X:=0
115420                    DO AD:=DPCCTAB(X) WHILE A><-1
115424                       IF A/\377=CLOGPAGE THEN
115430                          IF FPHYS+CA><D THEN EXITA FI    % WRONG SIZE
115435                          L+1; EXITA                      % OK
115437                       FI; X+2
115440                    OD; 304; EXIT                         % ERROR
115443                 ELSE                                     % SEGMENT
115444                    IF X:=SGADR.BPAGLINK=0 THEN CSEGN SH 10+305; EXIT FI  % ERROR, SEG. NOT IN MEMORY
115454                    T:=CORMBANK; *LDATX DPAGP
115456                    A=:CLOGPAGE=:CT
115460                    DO
115460                       *LDATX DPAGP
115461                       IF A><CLOGPAGE THEN CSEGN SH 10+306; EXIT FI    % ERROR, SEG. NOT CONTINUOUS
115470                       MIN CLOGPAGE; T:=CORMBANK; *LDXTX DPAGL
115473                    WHILE X><0
115474                    OD
115475                    IF CA+FPHYS><CT THEN CSEGN SH 10+307; EXIT FI    % WRONG PHYSICAL ADDRESS
115506                    L+1; EXITA                            % OK
115510                 FI
115510              FI
115510           OD; EXITA                                      % NOT SHARED AREA
115513    2F500: 0=:PARTINDX=:RETINDX=:PGFOUND; GO DO1; *)FILL
115527    DO1:   IF PARTINDX>=NPARTS SH 1 THEN GO FAR 55ERR FI
115535              "DPARTADDR"; X:=PARTINDX+A; T:=0; *LDDTX      % AD:=DPARTADDR(PARTINDX)
115542              A=:FPHYS:=D=:CURPAGE; X:=ECORMAP; 0=:CPEND
115547              T:=CORMBANK; *LDATX DPAGP
115551              IF A<<FPHYS GO IO1ND
115554              IF A<<CURPAGE THEN A=:CURPAGE FI
115560    DO2:      IF X<<CORMSTART GO FAR EDO2
115563                 T:=CORMBANK; *LDDTX DPGPR
115565                 X=:CMMENTRY
115566                 IF D=CURPAGE THEN
115571                    IF A/\377=0 OR A>>SGMAX GO NUSEPAGE
115576                    A=:CSEGN*5SEGSIZE+SEGSTART=:SGADR
115602                    IF A.FLAG BIT 5FIX GO NUSEPAGE
115606                    IF CPEND=0 THEN CURPAGE=:CPEND FI
115612                    X:=CMMENTRY; SGADR; CALL OUTLINK
115615                    T:=CORMBANK; *LDATX DPGPR
115617                    IF A BIT 5WIP THEN
115621                       SGADR.FLAG=:COLDFLAG BZERO 50K=:X.FLAG
115626                       X:=CMMENTRY; T:=SGADR; A:=1=:D; CALL XTOPOFF(ATRNSEG)
115634                       IF SGADR.FLAG BIT 5DEMAND THEN COLDFLAG=:X.FLAG FI
115642                    FI; X:=CMMENTRY; T:=0; SGADR; CALL XLINKOVER
115646                    X:=CMMENTRY; T:=CORMBANK; 160377; *STATX DPGPR
115652                    MIN PGFOUND
115653                    IF PGFOUND=NPAGES THEN
115657                       AD:=CCAREA; T:="DRETADDR"; X:=RETINDX+T; T:=0; *STDTX  %=:DRETADDR(RETINDX)
115665                       X:=PAGPN; A:=160000; T:=CORMBANK
115670                       DO WHILE X><0
```

```
115671                          *STATX DPGPR; LDXTX DPAGL
115673                       OD
115674                       PAGPN=:BREG.ZTREG; 0=:PAGPN
115700                       A:=RETINDX SHZ -1+1; GO FAR OKRET; *)FILL
115724                    FI
115724                    IF CURPAGE=FPHYS THEN
115730    TODO1:             AD:=CCAREA; T:="DRETADDR"; X:=RETINDX+T; T:=0; *STDTX  %=:DRETADDR(RETINDX)
115736                       X:=RETINDX+2=:RETINDX
115741                       IF A:=X SHZ -1>=MXMININTERVAL GO 55ERR
115746    IO1ND:             PARTINDX+2=:PARTINDX
115751                       GO FAR DO1
115752                    FI
115752    IO2ND:          CURPAGE-1=:CURPAGE
115755    IO3ND:          X:=CMMENTRY-4
115757                    GO FAR DO2
115760                 FI
115760    NOSEPAGE:    IF CPEND><0 THEN
115762                    AD:=CCAREA; A+1; T:="DRETADDR"; X:=RETINDX+T; T:=0; *STDTX  %AD=:DRETADDR(RETINDX)
115771                    X:=RETINDX+2=:RETINDX
115774                    IF A:=X SHZ -1>=MXMININTERVAL GO 55ERR
116001                    0=:CPEND
116002                 FI
116002                 IF CURPAGE=FPHYS GO IO1ND
116006                 X:=CMMENTRY; T:=CORMBANK; *LDATX DPAGP
116011                 IF A=CURPAGE GO IO2ND
116014                 IF A>FPHYS GO IO3ND
116017                 GO IO1ND                                                    ...
116020    EDO2:     IF CPEND><0 THEN CURPAGE+1=:CURPAGE; GO TODO1 FI
116026             GO FAR IO1ND
116027    *)FILL
116033
116033    55ERR: X:="PAGPN"; T:=0
116035           DO
116035              X=:D; T=:L; *LDXTX DPAGL
116040           WHILE X><0
116041              T:=CORMBANK; *LDATX DPGPR
116043              IF A/\377-377=0 THEN
116046                 A:=1; *STATX DPGPR; LDATX DPAGL
116051                 X:=:D; T:=L; *STATX DPAGL
116054                 X=:CMMENTRY; "XSGRT".BPAGLINK; A:=:D; A=:X.BPAGLINK
116061                 X:=A; T:=CORMBANK; A:=D; *STATX DPAGL
116065                 X:=CMMENTRY; T:=L
116067              FI
116067           OD
116070           IF "XSGRT".SEGLINK=0 AND X.BPAGLINK><0 THEN
116075              X:="BSEGLINK"; DO WHILE X.SEGLINK><-1; X:=A; OD
116104              "XSGRT"=:X.SEGLINK; -1=:"XSGRT".SEGLINK
116111           FI; GO FAR 5ERR
116112    *)FILL
116120
116120
116120    3F500: 0=:PARTINDX
116121           DO WHILE PARTINDX<NPARTS SH 1
116126              "DPARTADDR"; X:=PARTINDX+A; T:=0; *LDDTX       %AD:=DPARTADDR(PARTINDX)
116133              A=:FPHYS:=D=:CURPAGE
116136              X:="PAGPN"; T:=0
116140              DO
116140                 X=:PREVX; T=:PREVT; *LDXTX DPAGL
116143                 IF X=0 THEN CALL ERRFATAL FI
116146                 T:=CORMBANK; *LDATX DPAGP
```

```
116150              WHILE A><FPHYS
116153                 T:=CORMBANK
116154              OD
116155              X=:CBPGLINK
116156              DO
116156                 T:=CORMBANK; *LDATX DPAGP
116160              WHILE A><CURPAGE
116163                 T:=CORMBANK; *LDATX DPAGL
116165                 IF A=0 THEN CALL ERRFATAL FI
116167                 A=:X
116170              OD; T:=CORMBANK; *LDATX DPAGL; STZTX DPAGL
116174              T:=PREVT; X:=PREVX; *STATX DPAGL
116177              T:=CORMBANK; X:=CBPGLINK
116201              DO
116201                 A:=1; *STATX DPGPR; LDXTX DPAGL
116204              WHILE X><0
116205              OD; X=:D; "XSGRT".BPAGLINK; X:=:D; *STATX DPAGL
116213              CBPGLINK=:D.BPAGLINK
116216           OD; GO FAR OKRET;*)FILL
116225
116225    4F500: T:=CORMBANK; X:=ECORMAP; *LDDTX DPGPR
116230           IF ALPHYS>>D THEN A:=D=:ALPHYS FI
116235           NPAGES=:CSGNO
116237           IF A<=0 OR A>SGMAX GO 2ERR
116244           A*5SEGSIZE+SEGSTART=:SGADR; A.LOGADR SHZ -10=:NPAGES
116253           IF A=0 GO 2ERR
116254           IF X.FLAG=0 OR A BIT 5INHB OR A BIT 5DEMAND GO 2ERR
116262           IF A BIT 5FIX THEN
116264                 X:=X.BPAGLINK; D:=0
116266                 DO
116266                    T:=CORMBANK; *LDATX DPAGP
116270                    IF A<AFPHYS OR A>ALPHYS GO 2ERR
116276                    IF D=0 THEN
116300                       A=:D=:FPHYS
116302                    ELSE
116303                       IF A><D+1 GO 2ERR
116306                    FI; T:=CORMBANK; *LDXTX DPAGL
116310                 WHILE X><0
116311                 OD; A:=FPHYS; GO FAR OKRET
116314           FI; T:=CORMBANK; X:=ECORMAP; *LDDTX DPGPR
116317           IF AFPHYS+NPAGES>>D THEN 1; GO FAR 5ERR FI
116325           IF ALPHYS<<AFPHYS OR A-T+1<<NPAGES THEN 3; GO FAR 5ERR FI
116340           IF FIXPAGES+NPAGES>>FIXMAX THEN 3; GO FAR 5ERR FI
116347           CALL MBSREMOVE; 0=:NSHARED; GO FAR 500FIXC
116352    2ERR:  2; GO FAR 5ERR; *)FILL
116367
116367    5F500: T:=CORMBANK; X:=ECORMAP; *LDDTX DPGPR
116372           IF ALPHYS>>D THEN A:=D=:ALPHYS FI
116377           X:=0
116400           DO WHILE X><"NALME+NALME"
116403                 IF FXCTAB(X)=0 GO INDXFOUND
116405                 X+2
116406           OD; A:=2; GO FAR 5ERR
116411    INDXFOUND: X=:CTINDX; T:=CORMBANK; X:=ECORMAP; *LDDTX DPGPR
116415           IF AFPHYS+NPAGES-1>>D THEN 1; GO FAR 5ERR FI
116424           IF ALPHYS<<AFPHYS OR A-T+1<<NPAGES THEN 3; GO FAR 5ERR FI
116437           0=:NSHARED; GO FAR 500FIXC
116441    *)FILL
116447
116447    GSUBR: A:=1; T:=CORMBANK
```

```
116451          DO WHILE X><0
116452             X=:D; *STATX DPGPR; LDXTX DPAGL
116455          OD; "XSGRT".BPAGLINK; X:=D; *STATX DPAGL
116462          FXCTAB(CCTINDX); 0=:FXCTAB(X); X+1; 0=:FXCTAB(X); A=:"XSGRT".BPAGLINK
116471          IF X.SEGLINK=0 THEN
116473             BSEGLINK
116474             DO WHILE A.SEGLINK><-1 OD
116502             "XSGRT"=:X.SEGLINK; T=:"XSGRT".SEGLINK
116506          FI; EXIT
116507
116507    0F500: IF X:=CCTINDX>>="NALME+NALME" OR X BIT "0" THEN 3; GO FAR 5ERR FI
116517          T:=CORMBANK; X:=FXCTAB(X); *LDATX DPAGP
116522          IF A><AFPHYS THEN 3; GO FAR 5ERR FI
116527          CALL CSUBR; A:=0; GO FAR OKRET
116532
116532    7F500: IF NPAGES=0 THEN RTREF FI; A=:CCPRG
116536          A:=A-RTSTART=:D:=0; T:=5RTSIZE; *RDIV ST
116543          IF D><0 THEN 3; GO FAR 5ERR FI              % ILLEGAL RT-PROG.
116547          0=:CCTINDX
116550          DO WHILE X:=CCTINDX<<"NALME+NALME"
116554             X+1
116555             IF FXCTAB(X)=CCPRG THEN
116561                X-1; X:=FXCTAB(X); T:=CORMBANK; CALL CSUBR
116565             FI; CCTINDX+2=:CCTINDX
116570          OD; A:=0; GO FAR OKRET
116573    *)FILL
116604
116604    10F500: AFPHYS=:CLOGPAGE
116606          IF NPAGES>>SGMAX THEN 174; GO FAR 5ERR FI           % ILLEGAL SEGMENT NUMBER
116614          A*5SEGSIZE+SEGSTART=:SGADR                           % ADDR. OF SEG.TABLE ELEMENT
116617          IF A.FLAG=0 OR A BIT 5INHB THEN 2; GO FAR 5ERR FI    % SEGMENT NOT LOADED
116626          IF X.LOGADR SHZ -10<<CLOGPAGE THEN 174; GO FAR 5ERR FI % ILLEGAL PAGE NUMBER IN PAR.LISTE
116635          0=:CINDX; BREG.ZAREG=:CPHPAGE
116641          IF SGADR.FLAG NBIT 5FIX GO COKRET                    % SEGMENT NOT FIXED
116645          X.LOGADR/\377+CLOGPAGE=:CLOGPAGE; 0=:CCTINDX; 2=:CINDX
116654          T:=CORMBANK; IF X:=X.BPAGLINK=0 THEN 174; GO FAR 5ERR FI
116662          DO
116662             *LDATX DPAGP
116663             IF A=:D><CCTINDX+1 AND T:=CCTINDX><0 THEN 1=:CINDX ELSE A=:CCTINDX FI
116677             T:=CORMBANK; *LDATX DALOG
116701             IF A=CLOGPAGE THEN A:=D=:CPHPAGE FI
116706             T:=CORMBANK; *LDXTX DPAGL
116710          WHILE X><0
116711          OD
116712    COKRET: SGADR.FLAG=:BREG.ZDREG; CINDX=:X.ZTREG; CPHPAGE; GO FAR OKRET
116722    RBUS
116731
116731
116731    %-------------------------------------------------------------------------
116731    % 37.19       O U T L I N K
116731    %
116731    % SUBROUTINE TO REMOVE PAGES FROM A NON-DEMAND SEGMENT
116731    %
116731    % ENTRY:      X=CORMAP ADDRESS OF FIRST PAGE NOT TO REMOVE
116731    %             A=SEGMENT ADDRESS
116731    %
116731    SUBR OUTLINK
116731    INTEGER XREG,SGADR,SG1ADR:=XSGRT
116734    INTEGER POINTER LREG
116735
```

```
116735    OUTLINK: X=:XREG:=L=:"LREG"; A=:SGADR
116741             IF A.BPAGLINK=XREG OR X.FLAG BIT 5DEMAND OR X="XSGRT" GO OUT
116754             X.FLAG BZERO 50K=:X.FLAG
116757             DO WHILE X:=SGADR.BPAGLINK><XREG
116764                T:=CORMBANK; *LDATX DPGPR
116766                IF A BIT 5WIP THEN
116770                   A:=1=:D; T:=SGADR; CALL XTOPOFF(ATRNSEG)
116775                FI; T:=SG1ADR;; SGADR; CALL XLINKOVER
117000             OD
117001    OUT:     X:=XREG; GO LREG
117003
117003    RBUS
117007
117007
117007    %-----------------------------------------------------------------------------
117007    % 37.20          X L I N K O V E R
117007    %
117007    % SUBROUTINE TO MOVE A PAGE FROM ONE SEGMENT INTO ANOTHER SEGMENT
117007    %
117007    % ENTRY:         T=ADDRESS OF SEGMENT TO LINK PAGE INTO
117007    %                T=0 LINK TO N500
117007    %                T>0 AND T<SEGSTART THEN T-1=INDEX IN FXCTABLE
117007    %                T>=SEGSTART THEN T=ADDRESS OF ORDINARY SEGMENT
117007    %                X=ADDRESS OF CORMAP ENTRY
117007    %                A=ADDRESS OF SEGMENT TO LINK PAGE FROM
117007    %
117007    SUBR XLINKOVER
117007
117007
117007    INTEGER BREG,XREG,AREG,LSSEG,CURPAGE
117014    INTEGER POINTER LREG
117015
117015    XLINKOVER: T=:LSSEG; T:=:B; T=:BREG:=L=:"LREG"; A=:AREG; X=:XREG
117024             T:=0; X:=AREG+1
117027             DO T=:D; X=:L; *LDXTX DPAGL
117032             WHILE X><XREG
117035                T:=CORMBANK
117036             OD; A:=L=:AREG; T:=CORMBANK; *LDATX DPGPR
117043             A/\377*5SEGSIZE+SEGSTART=:LSSEG; AREG; *LDXTX DPAGL
117051             A:=:X; T:=D; *STATX DPAGL
117054             IF B>>SEGSTART THEN
117057                T:=5SEGSIZE; A:=B-SEGSTART; A=:D:=0; *RDIV ST
117065                A=:D:=FLAG/\177400+D; X:=XREG; T:=CORMBANK; *STATX DPGPR
117074                BPAGLINK; X=:BPAGLINK; *STATX DPAGL
117077             ELSE                                  % ALLOCATE NORD-500 MEMORY
117100                X:=XREG; T:=CORMBANK; *LDATX DPAGP
117103                A=:CURPAGE
117104                IF B=0 THEN "PAGPN" ELSE A:=B+"FXCTAB-1" FI
117112                X:=A; T:=0
117114                DO
117114                   X=:D; T=:L; *LDXTX DPAGL
117117                WHILE X><0
117120                   T:=CORMBANK; *LDATX DPAGP
117122                   IF A-CURPAGE>=0 THEN
117124                      X:=D; T:=L; *LDATX DPAGL
117127                      A=:D:=XREG; *STATX DPAGL
117132                      X:=A; T:=CORMBANK; A:=D; *STATX DPAGL
117136                      GO XOUT
117137                   FI
117137                OD; X:=D; T:=L; XREG; *STATX DPAGL
```

```
117144              X:=A; T:=CORMBANK; *STZTX DPAGL
117147    XOUT:     160000; *STATX DPGPR
117151            FI; IF LSSEG.BPAGLINK=0 AND X.SEGLINK><0 THEN CALL XTOPOFF(SREMOVE) FI
117160    OUT:    IF X:=LSSEG>=SEGSTART THEN
117164              IF X.FLAG NBIT 5DEMAND THEN A BZERO 50K=:X.FLAG FI
117171            FI; BREG=:B; X:=XREG; GO LREG
117175
117175    RBUS
117206
117206    *"8SIBA+8SIBX+8SIBM
"117206    %=============================================================================
117206    %              M A P S I B   -   M S I B B
117206    %
117206    % SIBAS COMMUNICATION MONITOR CALL
117206    %
117206    SUBR 3MAPSIB,3MSIBB
117206
117206    INTEGER ARRAY POINTER ACCTAB:=CCTAB
117207    DOUBLE ARRAY POINTER DACCTAB=ACCTAB
117207    DOUBLE ARRAY CRTCADDR(0)
117207             *0;0;0;0;0;0
117215    +"8SIBX+8SIBM
"117215
117215    INTEGER IMAPSIB:=0            % ><0 AFTER FIRST MAPSIB
117216
117216    3MAPSIB: IF BACKGROUND><0 THEN CALL ESCOFF FI
117221            IF SRTCSTAT><0 GO FAR M2ERR
117224            X:=SIBBDEVS(ZTREG)
117226            IF X:=X.RTRES=0 GO FAR M1ERR; IF X.STATUS NBIT 5WAIT GO FAR M2ERR
117234            IF IMAPSIB=0 THEN
117236               0=:SICCOUNT
117237               FOR SICCOUNT TO MXSIBAS DO
117243                  T:=SICCOUNT SH 11; A:=177000-T SHZ -12+100=:L
117252                  X:=0
117253                  DO ACCTAB(X) WHILE A><-1
117257                     IF A/\377=L THEN
117262                        AD:=DACCTAB(X); A:=0; AD SH 12
117265                        IF T:=SICCOUNT NBIT "0" THEN T:=1000; D+T FI
117272    INDO1:                X:=SICCOUNT+X; AD=:CRTCADDR(X); GO EFOR
117276                     FI; X+2
117277                  OD; A:=0=:D; GO INDO1
117303    EFOR:       OD
117307    OKINIT:     1=:IMAPSIB
117311            FI; X:=ZTREG+X; AD:=CRTCADDR(X); IF A=0 AND D=0 GO FAR EE174
117317            AD=:SIDRTC
117320            X:=ZXREG; OLDPAGE=:D; CALL DALTON; *BSET ZRO
117325            MLEV; *MST PIE
117327            *BSET ONE; LDA 1,X; BSET ZRO
117332            IF A=0 OR A>>177 GO FAR EE174; A=:SICCOUNT; GO L1; *)FILL
117354    L1:
117354    "ELIB CXCPU
117354            SICCOUNT SH 2=:L; ZXREG=:D; X:=SIAD1; T:=SIAD2; *BSET ONE; MOVAP
117365    "ELIB
117365    "ELIB CXCPU-,
117365            *BSET ZRO
117366    T:SIBAS:                      % LET'S ACTIVATE SIBAS AND WAIT FOR REPLY
117366            *IOF
117367            X:=SIBBDEVS(ZTREG)
117371            IF X.SIB500=0 THEN
117373               IF X:=X.RTRES=0 GO M1ERR; IF X.STATUS NBIT 5WAIT GO M1ERR
```

```
117400              A BZERO 5WAIT=:X.STATUS
117402            ELSE
117403              IF X.RTRES=0 GO M1ERR
117405              A=:D; *IOF
117407              CALL TOPOFF(FSEMA); GO M1ERR; CALL TOPOFF(MMBREACTIVATE); GO M1ERR
117415            FI; GO OVER; *)FILL
117420    OVER:   RTREF.STATUS BONE 5WAIT=:X.STATUS; -1=:SRTCSTAT
117426            "STUPR"; *IRW MLEVB DP
117430            MLEV; *MST PID; PION
117433            X:=SIAD2; T:=SIAD1; *LDATX 10
117436            IF A=0 OR A>>177 GO EE174; A=:SICCOUNT
117443
117443    @LIB CXCPU
117443            SICCOUNT SH 2=:L; SIDRTC; T:=ZDREG; *BSET ONE; MOVPA
117452    @ELIB
117452    @LIB CXCPU-,
117452            *BSET ZRO
117453    MXRET:  0=:ZAREG; MIN ZPREG; 0/\0; GO MXRET
117457    *)FILL
117462    EE174:  174; GO CERET
117464    M2ERR:  A:=-2; GO CERET
117466    M1ERR:  A:=-1
117467    CERET:  A=:ZAREG; *PION
117471            GO MXRET
117472
117472    $MSIBB: X:=SIBAPDEVS(ZTREG)=:D; 0=:SIB500
117476            *IOF
117477            IF X:=X.RTRES><0 AND X.STATUS BIT 5WAIT THEN A BZERO 5WAIT=:X.STATUS FI
117506            IF ZAREG=0 THEN RTREF.STATUS BONE 5WAIT=:X.STATUS; A:=0 ELSE A:=1 FI
117517            A=:D.SRTCSTAT; MIN ZPREG; 0/\0
117523            "STUPR"; *IRW MLEVB DP
117525            MLEV; *MST PIE; MST PID; ION
117531            GO RET
117532
117532    RBUS
117537
117537
117537    *)8NSDB
117537     %=============================================================================
117537    %       S Y M B O L I C   D E B U G G E R
117537    %
117537    *)9SLPL
117537    *  *<* MAXUS MAXUS MAXUS MAXUS MAXUS MAXUS MAXUS
117537    *)ZERO
117537    INTEGER ARRAY DBUGSEG(0); * *+MAXUS/
117540    INTEGER ARRAY DATSEGS(0)
117540    *)9RLPL

% @MAC
)9SCLC
117540  % PICK UP SEGMENT NUMBERS DEFINED ON SIN4-GEN
117540    8DB01+8DB02+8DB03+8DB04+8DB05+8DB06+8DB07+8DB08+8DB09
117540  SGNUM; )KILL TEMP;TEMP=SGNUM+1;)KILL SGNUM;SGNUM=TEMP
117541  )8NSDB
117541  )KILL TEMP
117541  )9RCLC
% *)9SLPL
)9SLPL117541    INTEGER ARRAY DBGSEGS(0); * *+MAXUS/
117542    INTEGER ARRAY OLDSEGS(0); * *+MAXUS/
117543    INTEGER ARRAY BPHANDLER(0); * *+MAXUS/
```

```
117544    INTEGER ARRAY REGBL(0); * *+MAXUS/
117545    INTEGER ARRAY APTNO(0); * *+MAXUS/
117546
117546    SUBR 3BRPNT,3DEBUG
117546
117546    INTEGER ARRAY RWPAR=?
117546    INTEGER BLCKNO=?, NWRD=?, FILNR=?, 64K=?
117546
117546
117546    3BRPNT: CALL FINDINDEX; GO FAR RETU
117550           RTREF.ACTSEG/\177400+DATSEG(CUIDX)=:T; DBUGSEG(X)=:RTREF.RSEGM
117561           CALL MMEXY; T=:OLDSEG(CUIDX)
117564           200=:D; CALL DALTON; *BSET ZRO
117570           X:=REGBL(CUIDX)
117572           *LDF 7,B; RDCR DT; BSET ONE; STF ,X; BSET ZRO
117577           *LDF 12,B; BSET ONE; STF 3,X; BSET ZRO
117603           *LDD 15,B; BSET ONE; STD 6,X; BSET ZRO
117607           OLDPAGE/\600 SH -7+1; *BSET ONE; STA 10,X; BSET ZRO
117616           OLDPAGE/\177174+200=:OLDPAGE; BPHANDLER(CUIDX)=:ZPREG
117625           GO FAR RETU
117626    *)FILL
117645
117645    3DEBUGGER: IF ZTREG>>5 GO FAR RETU
117651           A GOSW GETDSEG,PLAC1,FAR PSTART,FAR READLOC,FAR WRITLOC,RELDSEG
117660
117660    GETDSEG:
117660             FOR X:=0 TO NMAXUSERS-1 DO
117665               IF DBGPROGS(X)=RTREF GO SGFOUND
117671             OD
117673             FOR X:=0 TO NMAXUSERS-1 DO
117700               IF DBGPROGS(X)=0 AND DATSEG(X)><0 THEN RTREF=:DBGPROGS(X); GO SGFOUND FI
117707            OD; GO FAR RETU
117712    SGFOUND: X=:CUIDX
117713           RTREF.ACTSEG/\177400+DATSEG(CUIDX)=:T=:DBGSEGS(X)
117722           CALL MMEXY; T=:OLDSEG(X);T=:A; A/\377*5SEGSIZE+SEGSTART=:X
117731           IF X.LOGADR SHZ -10=200 THEN T:=100 ELSE T:=0 FI
117741           X.LOGADR/\300+T SH 1=:APTNO(CUIDX)
117747           RTREF.RSEGM=:DBUGSEG(CUIDX)
117753           MIN ZPREG; 0/\0; GO FAR RETU
117756    *)FILL
120002
120002    RELDSEG: CALL FINDINDEX; GO FAR RETU; 0=:DBGPROGS(X); MIN ZPREG; 0/\0; GO FAR RETU
120010
120010    PLAC1: CALL FINDINDEX; GO FAR RETU
120012           ZAREG=:FILNR; 0=:BLCKNO; 7=:NWRD; "BUFFR"=:RWPAR(2)
120022           "RWPAR"; *MON 117
120024           IF A><0 THEN A=:ZAREG; GO FAR RETU FI
120027           GO PLAC2; *)FILL
120035
120035    PLAC2: 0=:RTREF.RSEGM; CALL GETDATAFIELD
120040           5BCOM; T:="BSTATE"; CALL XSTDFADDR
120043           T:=FILNR; X:="BUFFR"; CALL 2BDBRECOVER; GO ERET
120047           A:=0; MIN ZPREG; 0/\0
120052    ERET:  A=:ZAREG; CALL GETDATAFIELD
120054           5BUSER; T:="BSTATE"; CALL XSTDFADDR
120057           DBUGSEG(CUIDX)=:RTREF.RSEGM; T:=DBGSEGS(CUIDX); CALL MMEXY
120066           GO FAR RETU
120067
120067
120067    INTEGER ARRAY RWPAR:=(FILNR,"0",0,BLCKNO,NWRD)
```

```
120074    INTEGER FILNR,NWRD,64K,BLCKNO
120100
120100
120100    PSTART: 200=:D; CALL DALTON; *BSET ZRO
120104            CALL FINDINDEX; GO FAR RETU
120106            ZXREG=:BPHANDLER(CUIDX); ZAREG=:REGBL(X)
120113            X:=ZAREG
120114            *BSET ONE; LDF ,X; BSET ZRO; STF 7,B
120120            *BSET ONE; LDF 3,X; BSET ZRO; STF 12,B
120124            *BSET ONE; LDF 6,X; BSET ZRO; STT 15,B; STA 16,B
120131            IF D=3 THEN T:=APTNO(CUIDX) ELSE T:=200 FI; T=:D
120141            T:=OLDSEG(CUIDX); 0=:RTREF.RSEGM; CALL MMEXY
120146            OLDPAGE/\177177\/D=:OLDPAGE; GO RETU
120153    *)FILL
120177
120177    READLOC:
120177    WRITLOC: CALL FINDINDEX; GO RETU; T:=OLDSEG(X); 0=:RTREF.RSEGM
120204            CALL MMEXY
120205            IF ZAREG><0 THEN T:=APTNO(CUIDX) ELSE T:=200 FI
120213            T=:D; CALL DALTON; *BSET ZRO
120216            X:=ZDREG
120217            IF ZTREG=3 THEN
120223               *BSET ONE; LDA ,X; BSET ZRO
120226               A=:ZAREG
120227            ELSE
120230               A:=ZXREG; *BSET ONE; STA ,X; BSET ZRO
120234            FI; T:=DBGSEGS(CUIDX); DBUGSEG(X)=:RTREF.RSEGM; CALL MMEXY
120242            MIN ZPREG; 0/\0; GO RETU
120245
120245    FINDINDEX: FOR X:=0 TO NMAXUSERS-1 DO
120252                  IF DBGPROGS(X)=RTREF THEN X=:CUIDX; EXITA FI
120260               OD; EXIT
120263
120263    RETU:   *IOF                    % WHEN CALLING BRELEASE FROM APPL. LEVEL
120264            X:=CURPROG; CALL BRELEASE; GO RET  % RELEASE SYNC. DATAFIELD
120267
120267    RBUS
120306
120306    *'8MOLI
120306     %=============================================================================
120306    %
120306    % MON LOGIN
120306    %               <LOG.DEV>        : LOGICAL DEVICE NUMBER OF THE TERMINAL
120306    %               <USER NAME>
120306    %               <PASSWORD>
120306    %               <PROJECT PASSWORD>
120306    %               <SUBSYSTEM NAME>
120306    %               <PASET PARAMETERS>
120306    %               <RETURNED STATUS>
120306    %
120306
120306    SUBR 3MLOGIN
120306
120306    INTEGER MLCPLIST=?
120306    GSX:    X:=MLCPLIST+A; *BSET ONE
120311            X.S0; *BSET ZRO
120313            EXIT
120314
120314    (MOVSTR: A=:MLICADDR+20=:MLIMXADDR
120317            DO WHILE X:=MLICADDR><MLIMXADDR
```

```
120323              *BSET ONE
120324              X.S0; *BSET ZRO
120326              IF A=:MLIWORD SHZ -10=15 OR A=##' GO OUT
120336              X:=MLICPNT; T:=B; *SBYT
120341              MIN MLICPNT; IF MLIWORD/\377=15 OR A=##' GO OUT
120352              X:=MLICPNT; T:=B; *SBYT
120355              MIN MLICPNT; MIN MLICADDR
120357           OD; EXIT
120361   OUT:    15; T:=B; X:=MLICPNT; *SBYT
120365           MIN MLICPNT
120366           EXITA
120367   *)FILL
120370
120370   INTEGER MLCPLIST
120371   3MLOGIN: ZAREG=:MLCPLIST
120373           MLEV; *MST PIE
120375           CALL ALTON; *BSET ZRO
120377           "S6"; CALL GSX; A=:MLIADDR
120402           DO; CALL LOGPH; IF A=0 OR D=0 GO EDEV
120407           IF A.TYPRING NBIT 5TERM AND A NBIT 5BAD GO EDEV
120415           IF X.RTRES><0 GO EOCCU
120417           X=:MLITERM
120420           IF UNAFLAG><0 AND RTRES><"STSIN" GO ESUNA
120426           MLISTRNG SH 1=:MLICPNT
120431           "S1"; CALL GSX; CALL CMOVSTR; GO ESTRNG
120435           "S2"; CALL GSX; CALL CMOVSTR; GO ESTRNG
120441           "S3"; CALL GSX; CALL CMOVSTR; GO ESTRNG
120445           "S4"; CALL GSX; CALL CMOVSTR; GO ESTRNG
120451           "S5"; CALL GSX; X:=MLIPASET+B; T:=12=:D; D BONE 16
120460           *BSET ONE; MOVB; JMP *; BSET ZRO
120464           MLICPNT=:MLIMXADDR; MLISTRNG SH 1=:MLICPNT
120471           *IOF
120472           IF MLITERM.RTRES><0 GO XEOCCU
120475           T:="FLAGB"; CALL XGTDFADDR; A BONE 5MLGIN; T:="FLAGB"; CALL XSTDFADDR
120502   *'8MOLI -8BACS
"120502            A:=X; *IRW MLEVB DX
120504           "CMBABPROC"; *IRW MLEVB DP
120506           MLEV; *MST PID; ION
120511           GO ENOFREE; *IOF
120513   *"8MOLI
"120513            CALL RTENTRY
120514           RTREF.STATUS BONE 5WAIT=:X.STATUS
120520           TTMR=:TMR; "RWAIT"; *IRW MLEVB DP
120524           MLEV; *MST PID
120526   RETU:   *ION
120527           X:=MLIADDR; MLIMSTATUS; *BSET ONE
120532           A=:X.S0; *IOF; BSET ZRO
120535           X:=RTREF; CALL BRELEASE; GO RET
120540   *)FILL
120563
120563   ESUNA: A:=-1; GO EFEL
120565   XEOCCU:
120565   EOCCU: A:=-1; GO EFEL
120567   EDEV:  A:=-1; GO EFEL
120571   XPEND:
120571           A:=-1; GO EFEL
120573   ESTRN: A:=-1; GO EFEL
120575   ENOFREE: A:=-1
120576   EFEL:   A=:MLIMSTATUS; GO RETU
120600
```

```
120600    RBUS
120600    *'
'120600
120600
120600    %=============================================================================
120600    % 34.27       G R T D A   -   G S G N O
120600    %
120600    % GRTDA: MONITOR CALL TO GET ADDR. OF RT-DESCRIPTION
120600    % GSGNO: MONITOR CALL TO GET SEGMENT NUMBER FROM SEGMENT NAME
120600    %
120600    SUBR 3GRTDA,3GSGNO
120600
120600    DISP 0; DOUBLE DOD1=D1,DOD2=D2; PSID
120600    DISP 0; REAL FS0=S0; PSID
120600    INTEGER CCRSG=?,CFLAG=?
120600
120600    3GSGNO: 5; GO FELLS
120602    3GRTDA: 7
120603    FELLS: A=:CFLAG; T=:D0
120605           MLEV; *MST PIE
120607           IF LOADI=0 GO FAR ERR
120612           0=:D1=:D2=:D3
120615           CALL ALTON; *BSET ZRO
120617           FOR X:=D0 TO D0+3 DO
120624                 RTREF=:D; A:=X; T:=1; CALL CHLIM; GO FAR ERR; *BSET ONE
120633                 T:=X.S0; *BSET ZRO
120635                 T=:D4
120636                 IF D4 SHZ -10/\177=##' GO OUT
120644                 A BZERO 6=:T
120646                 AD:=DOD1 SH 6; A=:D1; AD:=DOD2 SH 6
120653                 D+T; AD=:DOD2
120655                 IF D4/\177=##' GO OUT
120662                 A BZERO 6=:T
120664                 AD:=DOD1 SH 6; A=:D1; AD:=DOD2 SH 6
120671                 D+T; AD=:DOD2
120673           OD; GO OUT; *)FILL
120704    INTEGER CCRSG,CFLAG
120706    OUT:   RTREF.ACTSEG=:D4; X.RSEGM=:CCRSG; 0=:X.RSEGM; T:=5RTFIL; CALL MMEXY
120716           X:=6; CALL RALTON; X.S0; *BSET ZRO
120722           A=:D0; X+1
120724           DO WHILE D0><0
120726                 *BSET ONE
120727                 X.S4; *BSET ZRO
120731                 IF A/\7=CFLAG THEN
120735                    *BSET ONE
120736                    TAD:=X.FS0; *BSET ZRO
120740                    IF A-D2=0 AND D=A:=D3 AND T=A:=D1 GO FOUND
120750                 FI
120750                 X+7; D0-1=:D0
120754           OD; T:=D4; CCRSG=:RTREF.RSEGM; CALL MMEXY
120762    ERR:   -1=:ZAREG; GO RET
120765    FOUND: *BSET ONE
120766           IF CFLAG=7 THEN X.S5; T:=0 ELSE X.S3; T:=1 FI; *BSET ZRO
121000           A=:ZAREG; ZPREG+T=:ZPREG; CCRSG=:RTREF.RSEGM; T:=D4; CALL MMEXY
121011           GO RET
121012
121012    RBUS
121016
121016
121016    %=============================================================================
```

```
121016   % 34.28      G R T N A
121016   %
121016
121016   SUBR 3GRTNA
121016
121016   DISP 0; REAL FZTREG=ZTREG,FS0=S0; PSID
121016   INTEGER CCRSG
121017
121017   3GRTNA: IF LOADI=0 GO ERR
121021          MLEV; *MST PIE
121023          RTREF.ACTSEG=:D1; IF D0=0 THEN X=:D0 FI
121031          X.RSEGM=:CCRSG; 0=:X.RSEGM; T:=5RTFIL; CALL MMEXY
121036          X:=6; CALL RALTON; X.S0; *BSET ZRO
121042          A=:D2; X+1
121044          DO WHILE D2><0
121046                *BSET ONE
121047                X.S4; *BSET ZRO
121051                IF A/\37=27 THEN
121055                   *BSET ONE
121056                   X.S5; *BSET ZRO
121060                   IF A=D0 THEN
121063                      *BSET ONE
121064                      TAD:=X.FS0; *BSET ZRO
121066                      GO OUT
121067                   FI
121067                FI; X+7; D2-1=:D2
121073          OD
121074          TAD:=FZTREG; D:=0
121076   OUT:   TAD=:FZTREG; CCRSG=:RTREF.RSEGM; T:=D1; CALL MMEXY
121104          GO RET
121105   ERR:   0=:ZDREG; GO RET
121107
121107   RBUS
121115
121115   %=============================================================================
121115   %             3 S P L R E E
121115   %
121115   % THIS ROUTINE IS CALLED FROM SPLREE
121115   %
121115   SUBR 3SPLRE
121115
121115   DISP 26                          % D5+1
121115   INTEGER CBSGNO,CPOTADDR,CCFP,CCLP,CORSEGM,CCBMX,CCPOTADDR,CREBMAP,CCSGNO
121115   INTEGER NPAGES=CCBMX,PREVT=CCBMX,PREVX=CCPOTADDR,CURRX=CCSGNO
121115   INTEGER CFP1=D1,CLP1=D2,CFP2=D3,CLP2=D4
121115   DOUBLE DDDS1=CFP1,DDDS2=CFP2,DDDD0=D0,DDDD2=D2,DDDD4=D4,CCLMS=CCFP
121115   PSID
121115   DISP 100; DOUBLE D1PREV,D2PREV; PSID   % PREVIOUS LOGICAL ADDRESS AREAS
121115   DISP 0; TRIPLE 3BM0,3BM3; DOUBLE 2BM6; PSID
121115
121115   % COMPUTE ADDRESSES AND SIZES USED BY SEVERAL ROUTINES
121115   %
121115   GIVALUES: AD=:CCLMS; *BLDA 00 DA
121117             T:=D-A; AD SHZ -4; A+CREBMAP=:CCBMX
121124             AD SHZ 7; A/\170=:D
121127             CCFP SHZ -1; X:=CPOTADDR+A=:CCPOTADDR
121134             EXIT
121135
121135   % FOR ALL PAGES IN PREVIOUS LOGICAL ADDRESS AREA DO
121135   %       IF BIT OF CORRESPONDING PAGE IN BITMAP IS SET THEN
```

```
121135    %          SET CORRESPONDING POT-TABLE ENTRY EQUAL CURRENT REENTRANT SEGMENT
121135    %       FI
121135    % OD
121135    %
121135    INTEGER POINTER LREG=?
121135    INTEGER ENDINSTR(0); *BSKP ONE 170 DT
121136    INTEGER CINSTR(0); *BSKP ONE 00 DT
121137    STPOTTABLE: T:=L=:"LREG"; CALL GTVALUES
121142             T=:L; A:=CINSTR; D\/A
121145             DO WHILE L>>0                              % FOR ALL BITMAP LOCATIONS DO:
121147                IF T:=CCBMX.S0><0 THEN                  % IF UNEQUAL ZERO
121153                   *EXR SD                              % TEST EACH BIT
121154                   GO NOTSET
121155                   A:=CORSEG; *BSKP ONE SSK; SHA 10     % A = RSEGM READ FROM RT-D
121160                   A=:CCSGNO; X=:CCBMX:=CCPOTADDR       % X = ADDRESS OF POT-ENTRY
121163                   T:=SREBBANK; *LDATX                  % T = BANK NO. WHERE POT RESIDES; A = POT INFORMATION
121165                   IF K NBIT THEN                       % ACCORDING TO THE SETTING OF K-INDICATOR
121167                      A/\377
121170                   ELSE                                 % TAKE RIGHT- OR LEFTMOST BYTE
121171                      A/\177400
121172                   FI
121172                   A\/CCSGNO; *STATX                    % COMBINE WITH RSEGM-INFO AND UPDATE POT-ENTRY
121174                   *BSKP ZRO SSK
121175                   MIN CCPOTADDR; 0/\0                  % INCREMENT POINTER IN POT-ENTRY
121177    NOTSET:        L-1
121200                   T:=10; IF D+T>>ENDINSTR GO INELSE    % MODIFY BSKP INSTRUCTION TO TEST FOR NEXT BIT
121205                   *BSET BCM SSK                        % SWITCH VALUE OF K-INDICATOR
121206                ELSE
121207                   A:=D SHZ -3/\17; T:=20-A; L-T; T+1 SHZ -1
121217                   CCPOTADDR+T=:CCPOTADDR
121222    INELSE:        CINSTR=:D; K:="0"; MIN CCBMX         % MAKE READY TO TEST NEXT BITMAP WORD
121226                FI
121226             OD; GO LREG
121230
121230    INTEGER CBSET(0); *BSET ZRO 00 DA
121231    INTEGER LBSET(0); *BSET ZRO 170 DA
121232    INTEGER CLBSGNO
121233    INTEGER POINTER LREG
121234    MBCLBITS: T:=L=:"LREG"; CALL GTVALUES
121237              T=:L; A:=CBSET; D\/A
121242              DO WHILE L><0                             % FOR ALL PAGES IN LOGICAL AREA DO:
121244                 X:=CCPOTADDR; T:=SREBBANK; *LDATX      % READ A WORD FROM POT-TABLE
121247                 *BSKP ONE SSK                          % SKIP NEXT IF PAGE NO IS AN ODD NUMBER
121250                 *SHA ZIN SHR 10                        % SEGMENT NUMBER IN RIGHTMOST BYTE
121251                 IF A/\377-DO><0 THEN                   % IF EQUAL SEGMENT NO IN PARAMETERLIST
121254                    *LDATX; BSKP ZRO SSK; JMP RBY
121257                    A/\377\/CLBSGNO; GO FELLS
121262    RBY:            A/\177400\/CBSGNO
121264    FELLS:          *STATX
121265                    X:=CCBMX; X.S0; *EXR SD             %   SET THE RESPECTIVE BIT TO 1
121270                    A=:X.S0                             %   IN BITMAP
121271                 FI; T:=10
121272                 IF D+T>>LBSET THEN                     % IF LAST BIT IN THIS LOC. HAS BEEN SET
121276                     CBSET=:D; MIN CCBMX; 0/\0          %   MAKE READY FOR THE NEXT ONE
121302                 FI
121302                 *BSET BCM SSK                          % SWITCH VALUE OF K (INDICATES ODD OR EVEN PAGE NO)
121303                 IF K NBIT THEN MIN CCPOTADDR; 0/\0 FI; L-1
121310              OD; GO LREG
121312    *)FILL
121317
```

```
121317    TRIPLE ALLONES(0); *-1;-1;-1
121322
121322    CCHLIMS: D+A; IF C OR A>>200 OR D>>T GO CR15ERRD
121332             EXIT
121333
121333    CR15ERRD: DO; GO R15ERRD
121335
121335    3SPLRE: RTREF.SEGM SHZ -10+1=:CBSGNO                    % GET BACKGROUND SEGMENT NO
121342            T:=A SH 10=:CLBSGNO
121345            IF A*5SEGSIZE+SEGSTART><NSEGB GO RETSTUPR       % IF UNEQUAL NSEGB START SCANNING EXEC.QUEUE
121352            A:=RTREF-"BAK01"=:D:=0; T:=5RTSIZE; *RDIV ST    % INDEX IN RT-D TABLE, BAK01 = INDEX 0
121360            A*POTESIZE+SREBADDR=:CPOTADDR; X.RSEGM=:CORSEGM % CPOTADDR = POINTER TO POT-ENTRY, SAVE REENTR.
121365            X.RTDLGADDR+5BITMAP=:CREBMAP
121370            IF DO=0 AND CORSEGM=0 THEN
121374               X:=NSEGB; CBSGNO; CALL WSEGX                 % WRITE SHADOW SEGMENT BACK
121377            FI
121377            IF DO><0 THEN                                   % IF SEGMENT NUMBER UNEQUAL 0
121401               CALL LEGSEG; IF A NBIT 5SREEP GO CR15ERRD    % LEGAL SEGMENT ? DEMAND SEGMENT?
121404               IF A SHZ -11/\3><0 GO CR15ERRD               % RING ><0 ?
121407               IF X.LOGADR SHZ -6/\3<1 OR A>2 GO CR15ERRD   % PAGE TABLE MUST BE 1 OR 2
121420            FI
121420            AD:=DDDS1; CALL CCHLIMS; AD=:DDDS1              % LOGICAL AREA 1 TOO LARGE?
121423            AD:=DDDS2; CALL CCHLIMS; AD=:DDDS2              % LOGICAL AREA 2 TOO LARGE?
121426            IF D5><0 THEN                                   % IF POT-ENTRY FOR THIS PROGRAM IS TO BE CLEARE
121430               X:=CPOTADDR; A:=X+100=:L                     %    X POINTS TO START OF POT-ENTRY, L POINTS T
121434               CBSGNO SH 10\/CBSGNO=:D; T:=SREBBANK
121441               DO WHILE X><L                                %    FOR ALL BYTES IN THE POT-ENTRY
121443                  *STDTX; AAX 2                             %    FILL IN BACKGROUND SEGMENT NUMBER
121445               OD;
121446            ELSE
121447               IF CORSEGM><0 THEN
121451                  X:=CPOTADDR; T:=SREBBANK; *AAX D1PRE; LDDTX
121455                  CALL FAR STPOTTABLE
121456                  X:=CPOTADDR; T:=SREBBANK; *AAX D2PRE; LDDTX
121462                  CALL FAR STPOTTABLE
121463               FI
121463            FI
121463            TAD:=ALLONES=:CREBMAP.3BM0=:X.3BM3; AD=:X.2BM6
121470            IF DO><0 THEN
121472               AD:=DDDS1; CALL FAR MBCLBITS
121474               AD:=DDDS2; CALL FAR MBCLBITS
121476            FI; GO L1; *)FILL
121517
121517    INTEGER POINTER PPBPAG1:=BPAG1
121520    INTEGER CBSKP(0); *BSKP ZRO 00 DT
121521    L1:     AD:=DDDS1; T:=SREBBANK; X:=CPOTADDR; *AAX D1PRE; STDTX
121526            T:=100; A+T; D+T; AD=:DDDS1
121532            AD:=DDDS2; T:=SREBBANK; *AAX D2PRE-D1PRE; STDTX
121536            T:=100; A+T; D+T; AD=:DDDS2
121542            IF DO><0 THEN
121544               T:=0; X:=NSEGB+1   % BPAGLINK=1
121547               DO
121547                  T=:PREVT; X=:PREVX; *LDXTX DPAGL
121552               WHILE X><0
121553                  T:=CORMBANK; X=:CURRX; *LDATX DALOG
121556                  IF A>=CFP1 AND A<CLP1 GO COVERLAP
121564                  IF A<CFP2 OR A>=CLP2 GO NOVERLAP
121572    COVERLAP:     A-100; AD SHZ -4; A+CREBMAP=:X
121576                  AD SH 7; A/\170+CBSKP; T:=X.S0; *EXR SA
121603                  GO NOVERLAP
```

```
121604                  X:=CURRX; T:=CORMBANK; *LDATX DPAGL
121607                  A=:L:=PPBPAG1; *STATX DPAGL
121612                  A:=1; *STATX DPGPR
121614                  X=:PPBPAG1:=PREVX; T:=PREVT; A:=L; *STATX DPAGL
121621                  GO EDO
121622  NOVERLAP:       T:=CORMBANK; X:=CURRX
121624  EDO:         OD
121625               IF PPBPAG1><0 AND "XSGRT".SEGLINK=0 THEN
121632                  X:="BSEGLINK"; DO WHILE X.SEGLINK><-1; X:=A; OD
121641                  "XSGRT"=:X.SEGLINK; -1=:"XSGRT".SEGLINK
121646               FI
121646            FI; DO=:RTREF.RSEGM; GO RETSTUPR
121652  RBUS
121662
121662
121662
121662
121662  %============================================================================
121662  %              S T S L I C E
121662  %
121662  % ROUTINE TO TIMESLICE ND-100 PROCESSES
121662  %
121662
121662  SUBR STSLICE
121662
121662  INTEGER CINDEX=?,CTSLPROC=?,CTSLSTATUS=?
121662
121662  STSLICE:
121662          MLEV; *MCL PIE                      % STSLICE ENTRY POINT
121664          0=:CINDEX; X:=FPTSLICE
121666          DO WHILE X<<LPTSLICE
121671             IF X.WLINK=0 GO NXTSLPROG                    % PROGRAM NOT ACTIVE
121673             X=:B=:CTSLPROC
121675             IF TSLSTATUS(CINDEX) NBIT 5NOSLICE AND NBIT 5SPRF THEN
121703                A=:CTSLSTATUS
121704                IF A BIT 5ESCF THEN                       % WAITING FOR ESCAPE PRIORITY?
121706                   A BZERO 5ESCF BZERO 5BRKF=:CTSLSTATUS  % CLEAR ESCAPE AND BREAK
121711                   A SHZ -7CUTY/\7                        % FIND SLICE TYPE
121713                   TSLESCELEM(A); GO INBRKTEST            % GET ESCAPE ELEMENT
121716                FI
121716                IF A BIT 5BRKF THEN                       % WAITING FOR BREAK PRIORITY?
121720                   A BZERO 5BRKF=:CTSLSTATUS              % CLEAR BREAK FLAG
121722                   IF STATUS/\377<=TSLLOWLG THEN          % CAN PRIORITY BE INCREASED
121727                      CTSLSTATUS SHZ -7CUTY/\7            % FIND SLICE TYPE
121732                      TSLBRKELEM(A)                       % GET BREAK ELEMENT
121734  INBRKTEST:          A=:D:=CTSLSTATUS/\177740\/D=:CTSLSTATUS
121741                      DTIN2=:TSLNTIME(CINDEX)
121744                      GO SETALL
121745                   FI
121745                FI
121745  %             A:=TSLNTIME(CINDEX)-TSLIDLE=:TSLNTIME(X)-+DTIN2=:D:=0
121745                A:=DTIN2-TSLNTIME(CINDEX)=:D:=0
121752                T:=TSLTUNIT; *RDIV ST                     % COMPUTE CPU TIME USED
121754                IF A+TSLCOUNTA(X)<0 GO CONWAIT            % TIMESLICE FINISHED?
121756                DTIN2=:TSLNTIME(X)                        % YES, SET NEW CPU TIME
121760  FINNEW:       CTSLSTATUS/\37=:X                         % TIMESLICE ELEMENT
121763                A:=CTSLSTATUS/\177740\/TSLNEXTAB(X)       % FIND NEXT ELEMENT IN CHAIN
121766                A=:CTSLSTATUS/\37=:D
121771
121771  SETALL:       IF TSLTIMTAB(D)=:T-TSLHTIME>=0 THEN       % HASH ELEMENT?
```

```
121776                  DTIN2/\TSLHASHM+T                       % YES; HASH WITH TIME USED
122001               ELSE
122002                  A:=T
122003               FI; A-=:TSLCOUNTA(CINDEX)                  % SET TIMSLICE COUNTER
122006               STATUS/\177400\/TSLPRITAB(D)=:STATUS       % SET PRIORITY ON PROCESS
122013               X:=B; CALL FRWQU
122015               IF A><0 THEN B:=A; CALL TOWQU FI           % UPDATE EX-QUEUE
122020               CTSLSTATUS=:TSLSTATUS(CINDEX)
122023   CONWAIT:  FI; X:=CTSLPROC
122024   NXTSLPROG: X+5RTSIZE; MIN CINDEX
122026         OD; MLEV; *MST PIE
122031         CALL N500SCHEDULER                               % TIMESLISE ND-500 PROCESSES
122032         *IOF
122033         X:=RTREF; CALL FREXQ; 1=:MTOR
122037         "MONEN"; *IRW MLEVB DP
122041         MLEV; *MST PID; ION; JMP *                       % TERMINATE RTSLI (INSTEAD OF MON RTEXT)
122045
122045   INTEGER CINDEX,CTSLPROC,CTSLSTATUS
122050
122050   RBUS
122101   %=============================================================================
122101   %              T 3 R E P P      (TREPP  MON 332)
122101   %
122101   %       MONITORCALL TO DISABLE/ENABLE AND READ TERMINAL REPPORT
122101   %
122101   %       ENTRY: T = LOGICAL DEVICE NUMBER ( 1: OWN TERMINAL BACKGROUND)
122101   %               A = FUNCTION  = 0: DISABLE TERMINAL-REPPORT
122101   %                             = 1: ENABLE  TERMINAL-REPPORT
122101   %                             = 2: READ    TERMINAL-STATUS
122101   %       ERROR RETURN: A = ERROR NUMBER
122101   %       SKIP  RETURN: FUNCTION 0: OK
122101   %                     FUNCTION 1: OK
122101   %                     FUNCTION 2: OK  A = STATUS
122101   %                                         BIT  0: 1 - TERMINAL LINE DEAD
122101   %                                         BIT  1: 1 - LOGOUT WAITING FOR MON 0
122101   %                                         BIT  2: 1 - OVERRUN IN INPUT BUFFER
122101   %                                         BIT  3: 1 - PARITY ERROR IN INPUT
122101   %                                         BIT  4: 1 - FRAMING ERROR IN INPUT
122101   %       BITS USED IN FLAGB:
122101   %            BIT  0: 5TLREP
122101   %            BIT   : 5LOGOUT
122101   %            BIT 17: 6LSTA
122101   %       BITS USED IN TINFO:
122101   %            BIT 17: 5BFUL   - READ AND CLEAR
122101   %            BIT 16: 6PAER   - -----"-------
122101   %            BIT 15: 5FRER   - -----"-------
122101
122101   SUBR T3REPP
122101   INTEGER POINTER TTIF:=TTIFIELD,PBCHFLAG:=BCHFLAG
122103   T3REPP: IF BACKGROUND><0 AND ZTREG=1 THEN                % BACKGROUND OWN TERMINAL
122111              IF PBCHFLAG=1 GO DUMRE                       % FAKE ALWAYS OK IF BATCH
122115              X:=TTIF                                      % TTIFIELD IS TERM. DATAF.
122116           ELSE                                            % RT OR BACKGROUND DEV SPEC.
122117              IF ZTREG/\177700=100 GO ILLDV                % FILE IS ILLEGAL
122124              ZTREG; CALL LOGPH; IF A=0 GO ILLDV           % FIND DATAFIELD
122127              IF A.RTRES><RTREF GO NRESV                   % DEVICE MUST BE RESERVED
122134              IF X.TYPRING BIT 5TERM THEN
122137                 CALL XSETBFPAGE
122140              FI
122140           FI
```

```
122140          IF X.TYPRING BIT 5COM THEN A:="0"; T:=X.FLAGB; GO IGNLI FI
122146          IF A NBIT 5TERM AND A NBIT 5BAD THEN GO ILLDV FI
122153          IF ZAREG=0 THEN X.FLAGB BZERO 5TLREP=:X.FLAGB; GO DUMRE FI
122161          IF A=1 THEN X.FLAGB BONE 5TLREP=:X.FLAGB; GO DUMRE FI
122170          IF A=2 THEN
122173             A:="0"
122174             IF T:=X.FLAGB BIT 5LSTA THEN
122177                A BONE "0"                                    % LINE IS DEAD
122200             ELSE
122201                IF T BIT 5TLREP AND T BIT 5LOGOUT THEN        % TERMINATION STARTED
122205                   A BONE "0"   % LINE IS UP, BUT NOT RAPPORTED
122206                ELSE
122207                   A BZERO "0"  % LINE IS UP
122210                FI
122210             FI; *IOF
122211             T:=X.TINFO
122212             IF T BIT 5BFUL THEN T BZERO 5BFUL; A BONE 2 FI
122216             IF T BIT 5PAER THEN T BZERO 5PAER; A BONE 3 FI
122222             IF T BIT 5FRER THEN T BZERO 5FRER; A BONE 4 FI; T=:X.TINFO; *ION
122230  IGNLI:     IF T BIT 5LOGOUT THEN A BONE 1 ELSE A BZERO 1 FI % LOGOUT WANTED?
122235             GO RERET
122236          FI
122236          GO ILLFU
122237
122237  DUMRE:  A:="0"
122240  RERET:  MIN ZPREG; 0/\0
122242  RETU:   A=:ZAREG; GO RET
122244  NRESV:  A:=5; GO RETU                       % DEVICE NOT RESERVED
122246  ILLDV:  A:=240; GO RETU                     % ILLEGAL DEVICE TYPE
122250  ILLFU:  A:=201; GO RETU                     % ILLEGAL FUNCTION CODE
122252  RBUS
122262
122262  %-------------------------------------------------------------------------
122262  %  11.34      G D E V 3 T Y
122262  %
122262  %  MONITOR CALL TO GET DEVICE-TYPE AND ATTRIBUTES (MON 263)
122262  %
122262  %  ENTRY:          T-REG:    DEVICE NUMBER
122262  %                  A-REG:    0=INPUT, 1=OUTPUT
122262  %  SKIP RETURN:    T-REG:    DEVICE TYPE
122262  %                  A&D REG:  DEVICE ATTRIBUTES
122262  %  RETURN:         A-REG:    ERROR CODE
122262  %
122262  %  DEVICE TYPES RETURNED IN T-REG
122262  %       0 - UNSPECIFIED
122262  %       1 - TERMINAL
122262  %       2 - BACKGROUND ACCESS DEVICE (BAD)
122262  %       3 - COMMUNICATION CHANNEL
122262  %       4 - INTERNAL BLOCK DEVICE
122262  %       5 - FLOPPY-DISK
122262  %       6 - MAG-TAPE
122262  %       7 - MASS-STORAGE-FILE
122262  %
122262  %  ATTRIBUTES  RETURNED IN A&D REG
122262  %      BIT  0: INBT/OUTBT ALLOWED
122262  %      BIT  1: CONCT ALLOWED
122262  %      BIT  2: IOSET ALLOWED
122262  %      BIT  3: BLOCK CALLS ALLOWED
122262  %      BIT  4: CLEAR DEV ROUTINE AVAILABLE
122262  %      BIT  5: NO RESERVATION NECESSARY
```

```
122262   %      BIT  6: COSMOS REMOTE OPEN FILE
122262   %      BIT  7: NN-CHANNEL USED FOR REMOTE OPEN
122262
122262   SUBR GDEV3TY
122262   SYMBOL 9BTERM=1,9BBAD,9BCOM,9BIBDV,9BFLOP,9BMT,9BRFILE              % TYPES
122262   SYMBOL AIOBT,ACONCT,ATISET,AM144,ACLDV,ANORES,ACOSOP,ANNOP  % ATTRIBUTES
122262   INTEGER POINTER PTTNO:=TTNO,PBCHFLAG:=BCHFLAG, PTTIF:=TTIFIELD
122265
122265   GDEV3TY:
122265          IF BACKGROUND><0 AND ZTREG=1 THEN
122273             IF PBCHFLAG=0 THEN
122275                PTTNO
122276             ELSE
122277                X:=PTTIF
122300                IF ZAREG=0 THEN X.RIFIL ELSE X.DFOPP.ROFIL FI
122306             FI
122306          ELSE
122307             ZTREG
122310          FI
122310          CALL LOGPH
122311          IF T:=ZAREG><0 THEN D=:A FI
122315          IF A=0 GO ERR                                  % NO SUCH LOGICAL UNIT
122316          IF A.TYPRING BIT 5TERM THEN T:=9BTERM; GO ATRIB FI
122324          IF A BIT 5BAD    THEN T:=9BBAD;    GO ATRIB FI
122330          IF A BIT 5COM    THEN T:=9BCOM;    GO ATRIB FI
122334          IF A BIT 5IBDV   THEN T:=9BIBDV;   GO ATRIB FI
122340          IF A BIT 5FLOP   THEN T:=9BFLOP;   GO ATRIB FI
122344          IF A BIT 5MT     THEN T:=9BMT;     GO ATRIB FI
122350          IF A BIT 5RFILE THEN T:=9BRFILE; GO ATRIB FI
122354          T:=0
122355   ATRIB: T=:ZTREG:=0
122357          IF A BIT 5RFILE THEN
122361             A=:D; IF X.OFFLG BIT 5RCFIL THEN T BONE ACOSOP FI
122366             IF A BIT 5REMPO THEN T BONE ANNOP FI
122371             D=:A
122372          FI
122372          IF A BIT 5IOBT  THEN T BONE AIOBT FI
122375          IF A BIT 5CONCT THEN T BONE ACONCT FI
122400          IF A BIT 5ISET  THEN T BONE ATISET FI
122403          IF A BIT M144B  THEN T BONE AM144 FI
122406          IF A BIT 5CLDV  THEN T BONE ACLDV FI
122411          IF A BIT 5NORES THEN T BONE ANORES FI
122414
122414   UT:    T=:ZDREG; 0=:ZAREG; MIN ZPREG; 0/\0; GO RET
122421
122421   ERR:   33=:ZAREG; GO RET
122424   RBUS
122427
122427
122427   %=============================================================================
122427   %             3 G E T X T     (MON 334)
122427   %
122427   SUBR 3GETXT
122427   DISP 0; INTEGER SVACTSG=D0,SVRSGM=D1; PSID
122427   3GETXT:RTREF=:D; ZXREG; T:=100; CALL CHLIM; GO ERR  % CHECK USER'S BUFFER ADDRESS
122435          RTREF.ACTSEG=:SVACTSG; X.RSEGM=:SVRSGM       % SAVE USER'S SEGMENT
122442          MLEV; *MST PIE
122444          T:=ECODSEG; CALL MMEXY                       % GET ERROR-PROGRAM SEGMENT
122446          IF ZAREG<<ERO THEN
122452             ERSTB(A)
```

```
122454          ELSE
122455             IF A<<ERM THEN A-ERO ELSE A:=0 FI
122463             ERTAB(A)
122465          FI
122465    @LIB CXCPU
122465          T:="CBUFF"; D:=A; 100=:L; *MOVAN              % COPY ERROR-MESSAGE
122472          SVRSGM=:RTREF.RSEGM; T:=SVACTSG; CALL MMEXY   % UNSAVE USER'S SEGMENT
122477          OLDPAGE=:D; CALL DALTON; *BSET ZRO
122503          "CBUFF"=:D; T:=ZXREG; 100=:L
122510          *BSET ONE; MOVNA; BSET ZRO                     % MOVE TO USER'S BUFFER
122513          MIN ZPREG; 0/\0; GO RET
122516    @ELIB
122516    @LIB CXCPU-,
122516    ERR:    153=:ZAREG; GO RET                           % ILLEGAL ADDRESS REFS IN MONCALL
122521    *)FILL
122532    INTEGER ARRAY CBUFF(100)                             % BUFFER FOR ERROR-MESSAGE
122632    RBUS
122632
122632    %=============================================================================
122632    %              E X A B S        (MON 335)
122632    %
122632    % ENTRY:      X=WORKING AREA
122632    %              B=PARAMETER LISTE
122632    %
122632    SUBR 3EXABS
122632
122632    DISP 0; TRIPLE TRIP1=ZPREG,TRIP2=ZAREG,TRIP3=ZSREG; PSID
122632    DISP 4;  INTEGER A3FUNC;      PSID
122632    DISP 5;  DOUBLE A3MEMAD;      PSID
122632    DISP 20; DOUBLE A3MADDR;      PSID
122632    DISP 22; DOUBLE A3NWRD;       PSID
122632    DISP 24; INTEGER A3RETADDR;   PSID
122632    DISP 25; INTEGER A3PLIST;     PSID
122632
122632    INTEGER SVX=?
122632
122632    3EXABS:IF X.OLDPAGE NBIT "1" THEN
122635              X=:B; -1=:ZAREG; GO RETSTUPR              % ERROR, PROG NOT ON RING 2
122641          FI; MLEV; *MCL PIE
122643          X=:SVX
122644          FOR X:="FPL3ABS" STEP 31 TO "LPL3ABS"-31 DO      % FIND PARAMETER FIELD
122651             IF X.RTRES=0 OR A=RTREF GO FOUND
122656          OD
122660          IF X.RTRES=0 THEN X=:B:="DUMMY"; CALL BRESERVE; FI  % RESERVE QUEUEING SEMAPHORE FOR DUMMY
122665          X:=RTREF; CALL FREXQU; CALL TOWQU              % WAIT FOR ABSTR PARAMETER LISTE
122670          SVX=:B; ZPREG-1=:ZPREG; GO RETSTUPR; *)FILL
122706
122706    INTEGER SVX
122707    FOUND: X=:B:=RTREF; CALL BRESERVE                    % RESERVE ABSTR PARAMETER LISTE
122712          IF A<0 THEN CALL ERRFATAL FI; X:=SVX
122715          TAD:=X.TRIP1=:TRIP1:=X.TRIP2=:TRIP2:=X.TRIP3=:TRIP3
122723          X:=:B=:SVX:=RTREF; CALL BRELEASE             % RELEASE ORIGINAL WORKING FIELD
122727          SVX=:B; MLEV; *MST PIE
122733          OLDPAGE=:D; CALL DALTON; *BSET ZRO            % SET ALT.PIT=USER'S.ALT.PIT
122737          X:=ZAREG; *BSET ONE                          % SET UP ABSTR PARAMETER LISTE
122741          X:=X.S0; X.S0; *BSET ZRO
122744          A=:A3FUNC; X:=ZAREG; *BSET ONE
122747          X:=X.S1; X.DS0; *BSET ZRO
122752          AD=:A3MEMAD; X:=ZAREG; *BSET ONE
122755          X:=X.S2; X.DS0; *BSET ZRO
```

```
122760          AD=:A3MADDR; X:=ZAREG; *BSET ONE
122763          X:=X.S3; X.DS0; *BSET ZRO
122766          AD=:A3NWRD; X=:A3RETADDR
122770          T:=ZTREG; "A3PLIST"+B; *MON 2ABST
122774          A=:ZAREG; X:=A3RETADDR; A3NWRD; *BSET ONE
123000          AD=:X.DS0; *BSET ZRO
123002          MLEV; *MCL PIE
123004          IF "LPL3ABS".RTRES="DUMMY" THEN
123011             IF X.BWLINK><X THEN
123014                X:=:B; X=:SVX:=A; CALL FRWQU; CALL TOEXQU; SVX=:B
123023             FI
123023          FI; X:=RTREF; CALL BRELEASE
123025          GO RET
123026
123026    RBUS
123041
123041    *"8ESCX
"123041    %=============================================================================
123041    %               E L O N 3     E L O F 3
123041
123041    % MONITOR CALLS TO TURN ON OFF DELAYED ESCAPE HANDLING
123041    %
123041    % ELON3  (ELON  - MON 302)    - NORMAL  ESCAPE/LOCAL DETECTION
123041    % ELOF3  (ELOFF - MON 303)    - DELAYED ESCAPE/LOCAL DETECTION
123041    SUBR ELON3,ELOF3,T207RET
123041    INTEGER POINTER TTIF:=TTIFIELD
123042    ELON3:  IF TTIF=0 THEN 33; GO RETU; FI
123046            IF A.TYPRING NBIT 5TERM AND A NBIT 5BAD AND A NBIT 5COM THEN
123056               240; GO RETU
123060            FI
123060            IF X.FLAGB BIT 5WLOC THEN
123063                  IF A BIT 5LCHAR THEN
123065                     X.LUSADD=:ZPREG         % RETURN IN LOCAL CHAR HANDLING
123067                     GO ESON
123070                  FI
123070            FI
123070            IF A BIT 5WESC THEN
123072                  X:=:B; GO T2P07            % ESCAPE RESPONSE IF TAD
123074    T207RET:      X:=:B                      % T2P07 ALWAYS RETURNS TO T207RET
123075                  IF X.FLAGB BIT 6USESC THEN
123100                     X.EUSADD=:ZPREG         % RETURN IN USER ESCAPE HANDLING
123102                     GO ESON
123103                  ELSE
123104                     "XBRTWT"                % SYSTEM ESCAPE HANDLING
123105                     *IOF; IRW MLEVB DP
123107                     MLEV; *MST PID
123111                  FI
123111            FI
123111            MIN ZPREG; 0/\0
123113    ESON:   X.FLAGB BZERO 5ESCLOFF; GO OUT
123116
123116    ELOF3:  CALL GET0; IF TTIF=0 THEN 33; GO RETU; FI
123123            IF A.TYPRING NBIT 5TERM AND A NBIT 5BAD AND A NBIT 5COM THEN
123133               240; GO RETU
123135            FI
123135            X.FLAGB BONE 5ESCLOFF; MIN ZPREG; 0/\0; GO OFOUT
123142    OUT:    A BZERO 5WLOC BZERO 5WESC
123144    OFOUT:  A=:X.FLAGB; A:=0; *ION
123147    RETU:   A=:ZAREG; GO RET
123151    RBUS
```

```
123156
123156   %=============================================================================
123156   %              E L O 3 F   D L O F 3
123156
123156   % MONITOR CALLS TO ENABLE/DISABLE LOCAL FUNCTION
123156   %
123156   % ELO3F (ELOFU - MON 276)    - ENABLE  LOCAL FUNCTION
123156   % DLOF3 (DLOFU - MON 277)    - DISABLE LOCAL FUNCTION
123156   SUBR ELO3F,DLOF3
123156   INTEGER POINTER TTIF:=TTIFIELD
123157   ELO3F: K:=1; GO FELLS
123161   DLOF3: K:="0"
123162   FELLS: IF TTIF=0 THEN 33; GO RETU; FI
123166          IF A.TYPRING NBIT 5TERM AND A NBIT 5BAD THEN
123174             240; GO RETU
123176          FI
123176          IF K THEN
123200             ZAREG=:X.LUSADD; *IOF
123203             X.FLAGB BONE 5LCHAR=:X.FLAGB; *ION
123207          ELSE
123210             *IOF
123211             X.FLAGB BZERO 5LCHAR BZERO 5WLOC=:X.FLAGB; *ION
123216          FI
123216          MIN ZPREG; 0/\0; A:=0
123221   RETU:  A=:ZAREG; GO RET
123223   RBUS
123225
123225   %=============================================================================
123225   %              E U S E 3   D U S E 3
123225
123225   % MONITOR CALLS TO ENABLE/DISABLE USER ESCAPE HANDLING
123225   %
123225   % EUSE3 (EUSEL - MON 300)    - ENABLE  USER ESCAPE HANDLING
123225   % DUSE3 (DUSEL - MON 301)    - DISABLE USER ESCAPE HANDLING (NORMAL SINTRAN MODE)
123225   SUBR EUSE3,DUSE3
123225   INTEGER POINTER TTIF:=TTIFIELD
123226   EUSE3: K:=1; GO FELLS
123230   DUSE3: K:="0"
123231   FELLS: IF TTIF=0 THEN 33; GO RETU; FI
123235          IF A.TYPRING NBIT 5TERM AND A NBIT 5BAD THEN
123243             240; GO RETU
123245          FI
123245          IF K THEN
123247             ZAREG=:X.EUSADD; *IOF
123252             X.FLAGB BONE 5USESC=:X.FLAGB; *ION
123256          ELSE
123257             *IOF
123260             X.FLAGB BZERO 5USESC BZERO 5WESC=:X.FLAGB; *ION
123265          FI
123265          MIN ZPREG; 0/\0; A:=0
123270   RETU:  A=:ZAREG; GO RET
123272   RBUS
123274
123274   *'BIBRS
"123274   %=============================================================================
123274   %              I B R 3 S
123274   %
123274   % MONITOR CALL TO RETURN NUMBER OF CHARACTERS IN THE INPUT-BUFFER
123274   % IF BREAK STRATEGY APPLIES TO THE DEVICE, NUMBER OF CHARACTERS
123274   % UNTIL BREAK CONDITION IS ALSO RETURNED.
```

```
123274   % MON 313
123274   % ENTRY:        T = DEVICE-NUMBER
123274   % SKIP-RETURN:  A = NUMBER OF CHARACTERS IN BUFFER
123274   %                X = BREAK CHARACTER NUMBER
123274   % NOSKIP:       A = ERROR CODE
123274
123274   SUBR IBR3S,T206RET
123274   INTEGER POINTER PTTNO:=TTNO,PBCHFLAG:=BCHFLAG
123276   INTEGER BREG=?,BCOUNT=?,BHENTE=?,BBUFS=?
123276   @LIB CXCPU
123276   INTEGER IPIT3:=177000+5UBFPAGE+5UBFPAGE
123277   @ELIB
123277   IBR3S: CALL GZTREG; GO FAR ERR; IF A=0 GO FAR ERR; *IOF
123304          A:=:B=:BREG
123306          IF TYPRING BIT 5TERM THEN                    % TERMINAL
123311   @LIB CXCPU
123311             A;=TDFPHPAGE=:D:=162000; X:=IPIT3; T:=0; *STDTX
123317             TDFLGADDR/\1777+"5UBFPAGE*2000"=:B
123323   @ELIB
123323             0=:BCOUNT
123324             IF BRKTAB><0 THEN                                    % BREAK APPLIES
123326                IF BHOLD><0 THEN
123330                   IF DFLAG NBIT 5ECHO THEN                      % PAST BREAK IN DRIVER
123333                      HENTE=:BHENTE
123335                      IF RSISTE=-1 THEN                          % HENTE PAST FIRST BREAK
123341   CHCHR:                1=:BCOUNT
123343   @LIB CXCPU
123343                         A:=B/\176000+BUFST=:BBUFS
123347   @ELIB
123347   @LIB CXCPU-,
123347                         FOR X:=BCOUNT TO BHOLD DO; X=:BCOUNT    % SEARCH FOR BREAK
123354                            T:=BBUFS; X:=BHENTE
123356   @LIB CXCPU
123356                            *LBYT
123357   @ELIB
123357   @LIB CXCPU-,
123357                            X+1 IF X=MAX THEN X:=0 FI; X=:BHENTE
123365                            X:=BRKTAB; CALL VSXGETBIT; IF A<0 GO RETTR
123370                         X:=BCOUNT; OD
123373                         0=:BCOUNT
123374                         IF BRKMAX><0 THEN
123376                            IF BHOLD+NCBRK>=BRKMAX THEN
123403                               BRKMAX-NCBRK=:BCOUNT
123406                            FI
123406                         FI
123406                      ELSE
123407                         IF BHENTE=RSISTE GO CHCHR
123413                         DO                                      % BREAK FROM DRIVER
123413                            A+1; IF A=MAX THEN A:=0 FI           % FIND NUMBER
123420                            MIN BCOUNT; IF A=RSISTE GO RETTR
123424                         OD
123425             FI;FI;FI;FI
123425   RETTR:    A:=BHOLD
123426   @LIB CXCPU
123426             X:=IPIT3; T:=0; *STZTX
123431   @ELIB
123431             GO RETU; *)FILL
123440
123440   INTEGER BREG,BCOUNT,BHENTE,BBUFS
123444
```

```
123444          FI
123444          IF TYPRING BIT 5COM THEN                    % COMMINICATION CHANNEL
123447             IF BRKTAB><0 THEN
123451                BYTS-=:BCOUNT
123454             ELSE
123455                0=:BCOUNT
123456             FI
123456             A:=BYTS-; GO RETU
123461          FI
123461          IF TYPRING BIT 5BAD THEN                    % BACKGROUND ACCESS DEVICE
123464             X:=BREG; GO T2C06                        % RETURN IN BISIZ IF TAD INCLUDED
123466          FI
123466    T206RET;
123466          IF TYPRING BIT 5IOBT THEN
123471             0=:BCOUNT; A:=BHOLD; GO RETU
123474          FI
123474
123474    ERRB:   BREG=:B
123476    ERR:    240=:ZAREG; GO RET
123501    RETU:   T:=BREG=:B; T:=BCOUNT
123504          A=:ZAREG; T=:ZXREG; MIN ZPREG; 0/\0; GO RET
123511    RBUS
123514    *"
"123514
123514    %=============================================================================
123514    % 34.3        M C 1 4 4
123514    %
123514    %
123514    % CALL MAGTP(FUNC,CORAD,LOG.DEV,MAXWORDS,READWORDS)
123514    % ENTRY: B=DF-DATAFIELD
123514    %        PARAMETERS ARE IN DATAFIELD
123514    % EXIT: A=0 IF OK
123514    %
123514    SUBR PT3MC144,RMAGT
123514
123514    % VARIABLES IN I/O DATAFIELD:
123514
123514    DISP 25; INTEGER CASUN; PSID              % DEVICE UNIT NUMBER
123514
123514    % VARIABLES IN "DF" DATAFIELDS:
123514
123514    DISP 35
123514       INTEGER DMLOG                  % LOGICAL DEVICE NUMBER OF MASS STORAGE DEVICE
123514       INTEGER DMFLD                  % DATA FIELD ADDRESS OF MASS STORAGE DEVICE
123514       INTEGER IUNIT                  % DEVICE UNIT NUMBER
123514       INTEGER MDBUF                  % DEVICE BUFFER ADDRESS: LEAST SIG. 16 BITS
123514       INTEGER CMDBUF                 % DEVICE BUFFER ADDRESS: LEAST SIG. 16 BITS
123514                                      % WILL BE INCREMENTED WITH 1K FOR EACH K WORDS
123514                                      % TO READ/WRITE
123514       INTEGER MDFI                   % DEVICE BUFFER HEADER ADDRESS
123514       INTEGER FIMAX                  % NO. OF WORDS TO READ OR WRITE IF FLOPPY DISC
123514       INTEGER SVXERR=DMLOG           % TEMPORARLY SAVING OF X-REGISTER
123514       INTEGER TSAVE=MDBUF            % TEMPORARLY SAVING OF T-REGISTER
123514    PSID
123514
123514    %       DISPLACEMENTS IN INTERNAL BLOCK DEVICE DATAFIELDS.
123514
123514    DISP -3
123514       INTEGER DBLDN              % LOGICAL DEVICE NUMBER OF DBH RESERVED FOR THIS IBD. (ONLY FIRST).
123514       INTEGER INWORDS            % NO  OF WORDS IN BUFFER
```

```
123514      INTEGER IDFOPP              % ADDRESS OF OUTPUT DATAFIELD
123514    PSID
123514    DISP 13
123514       INTEGER BREGC
123514    PSID
123514
123514    INTEGER POINTER OFLD1=?,OFLD=?  % OPEN FILE TABLE LOCK
123514    INTEGER POINTER OFLD3:=OFLCK
123515
123515    PT3MC144:
123515           IF IOLOG>="RDVLO" AND <"RDVHI" THEN   % REMOTE PERIPHERAL?
123524              IOLOG-"RDVDF"=:IOLOG; CALL LOGPH; IF =0 THEN D=:A; FI; A=:DMFLD
123533              IF =0 THEN A:=174; GO FAR ERR; FI
123536              IF A.OFNBR<100 OR >177 THEN A:=201; GO FAR ERR; FI; GO RMAG
123551           FI
123551           IOLOG; CALL LOGPH; IF =0 THEN D=:A FI
123555           IF =0 OR A.TYPRING NBIT M144B THEN 174; GO FAR ERR FI
123564           IF A BIT 5COM THEN CALL COMBLOCK FI                  % 5COM: COMMUNICATION CHANNEL
123567           IF A BIT 5HDMA OR A BIT 5IBDV THEN                   % 5HDMA: BSC ON DMA, HDLC.
123573              IF IFUNC=1 OR =51 OR =52 OR =55 THEN D=:X FI      % 5IBDV: INTERNAL BLOCK DEVICE: X-REG POINTS TO
123611           FI
123611           IF X.RTRES><0 AND A><SSREF GO FAR E205
123616           X.CASUN=:IUNIT; X.CLOGDV=:DMLOG; CALL LOGPH
123623           IF A = 0 THEN A:=ER109; GO FAR ERR FI                % "ILLEGAL PARAMETER".
123626           A=:DMFLD
123627           IF A.TYPRING BIT 5IBDV THEN
123633    %
123633    %           INTERNAL BLOCK DEVICE. CHECK THAT A DMA DEVICE BUFFER HAS
123633    %           ALREADY BEEN OBTAINED AND IF NOT GET ONE.
123633    %
123633              X=:CMDBU; T:=X.DBLDN
123635              IF T = 0 THEN
123637                 A:=IMAXW; T:=IOLOG; CALL G3IBUF; GO FAR ERRBU
123643                 T=:MDFI; A:=D=:MDBUF; X=:D
123647                 A:=IMAXW+1777 /\ 177000=:T.CPAG2              % BUFFER OBTAINED FOR UP TO THIS NUMBER OF WORD
123654                 A:=X.LNUMB=:D.DBLDN                           % SAVE LDN OF DBH IN INPUT PART OF IBD DATAFIEL
123657                 0=:X.BREGC                                    % NEW DEVICE BUFFER; NO DATA IN IT.
123660              ELSE
123661                 A:=T; CALL XLOCK; A:=T                        % LOCK THE DBH
123664                 CALL LOGPH; A=:X=:MDFI; A:=X.BUFFER=:MDBUF    % LEAST SIG WORD OF DB ADDRESS.
123671                 IF A:=IMAXW > T:=X.CPAG2 THEN
123675                    A:=ER206; GO FAR ERR                       % TRYING TO READ OR WRITE TOO MANY WORDS IN THE
123677                 FI
123677              FI
123677              X:=CMDBUF; A:=MDBUF=:CMDBUF                      % X-REG NOW POINTS AT RIGHT DATFIELD.
123702              A;=OFLD3; CALL XUNLOCK
123704              GO FAR M144X
123705    *)FILL
123725           FI     % EG. ELSE
123725    RMAG:          IOLOG; CALL LOGPH
123727                 IF A><0 THEN IF A.RTRES><0 AND A><SSREF GO FAR E205 FI
123736                 IF D><0 THEN IF D.RTRES><0 AND A><SSREF GO FAR E205 FI
123746    *'8MDIR 99MGT
'123746           IF IFUNC<5 OR =25 OR =43 OR =44 OR =52 OR =55 THEN
123771              IMAXW
123772           ELSE
123773              IF A=26 OR A=27 THEN IMAXW+1 SHZ -1 ELSE A:=0 FI
124006           FI
124006           A=:D                                                 % NO. OF WORDS
124007           IF DMFLD.TYPRING BIT 5HDMA THEN
```

```
124013              IF BREGC=0 THEN                                    % DEVICE BUFFER IS PRESENT
124015                 T:=IOLOG; A:=D; CALL G3BUFF; GO ERRBU
124021                 A:=D=:MDBUF=:CMDBUF; T=:MDFI; 1=:BREGC
124027                 IOLOG; CALL LOGPH; D=:X                         % GET OPPOSIT DATAFIELD
124032                 X.DFDEV; CALL LOGPH; A=:X
124035                 IF X=B THEN                                     % THIS IS THE SAME.
124037                    IOLOG; CALL LOGPH; A=:X                      % GET OPPOSITE
124042                    X.DFDEV; CALL LOGPH; A=:X
124045                 FI
124045                 CMDBUF=:X.CMDBUF=:X.MDBUF
124050                 MDFI=:X.MDFI; 1=:BREGC
124054              ELSE
124055                 CMDBUF=:MDBUF                                   % DEVICE BUFFER OK
124057              FI
124057           ELSE
124060              T:=IOLOG; A:=D; CALL G3BUFF; GO ERRBU
124064              A:=D=:MDBUF=:CMDBUF; T=:MDFI
124070           FI
124070           IF DMFLD.OFFLG BIT 5RCFIL AND X.TYPRING BIT 5RFILE GO M144X
124077           OFLD1; CALL XUNLOCK; GO M144X
124102     *)FILL
124106
124106     INTEGER POINTER OFLD1:=OFLCK
124107
124107     ERRAD: 153; GO FAR RETU
124111     ERRBU: 172; GO ERR
124113     E201:  201; GO ERR
124115     E205:  205
124116     ERR:   GO FAR RETU2
124117
124117     M144X: IF DMFLD.TYPRING BIT 5FLOP THEN
124123              IFUNC/\77+"SFTIM"=:X; T:=0; *LDATX      % FLOPPY/STREAMER FUNCTION DESCRIPTION (IN POF)
124131              IF A BIT 3FRESTRICTED THEN
124133                 IF SSREF.ACTPRI BIT 5BACKR OR A/\3 < 2 GO E201  % ONLY LEGAL FOR RT-P ON RING 2
124143              FI
124143           FI
124143           IF IFUNC=20 OR =24 OR =57 OR =64 OR =65 OR =70 THEN
124166              CALL CALABSTR; GO FAR RETU; *)FILL
124200           FI
124200           IF A=42 OR A=46 OR =56 OR =63 THEN         % READ FORMAT/READ DISC ADDR.+ND852 FUNCTIONS
124214              T:=1; SSREF=:D; "IRETW"; CALL CHLIM; GO ERRAD
124222              CALL CALABSTR; X:=B; CALL GAPIT
124225              MDFI.DBLOCK; X:="IRETW"
124230              CALL DALTON; A=:X.SO; CALL ALTOFF; *BSET ONE
124234              GO FAR RETU1
124235           FI
124235     @ICR
124235           IF >4 AND A><43 AND A><44 AND A><25 AND A><52
124235             AND A><55 AND A><26 AND A><27 AND A><36 THEN
124235     @CR;
124270              CALL CALABSTR
124271     %
124271     %       CLEAR DEVICE; BLOCK INTERNAL DEVICES ONLY.
124271     %
124271              IF DMFLD.TYPRING BIT 5IBDV AND IFUNC=21 THEN
124301                 A:=X.DBLDN; CALL LOCK                           % LOCK DEVICE BUFFER HEADER.
124303                 T:=IOLOG; CALL R3IBUFF; CALL ERRFATAL
124306                 0=:X.DBLDN; 0=:X.IDFOPP.BREGC
124311              FI
124311              GO FAR RETU1
```

```
124312    *)FILL
124323            FI
124323
124323    %       READ OR WRITE FUNCTIONS:
124323
124323    L1:     IF IFUNC=2 OR=26 OR=27 THEN T:=IMAXW+1 SHZ -1 ELSE T:=IMAXW FI
124342            SSREF=:D:=ICORAD
124345            CALL CHLIM; GO FAR ERRAD                    % ILL. USER-BUFF-ADDR
124347            IF MTFLG><0 THEN
124351               T:=1; "IRETW"
124353               CALL CHLIM; GO FAR ERRAD                 % CHECK ADDRESS OF IRETW
124355            FI
124355            CALL BLOCK
124356            IF IFUNC><0 AND A><2 AND A><4 AND A><43 AND A><25 AND A><26 AND A><36 GO FAR L3    % READ
124402               IF DMFLD.TYPRING BIT 5FLOP THEN IMAXW=:FIMAXW FI
124410               CALL CALABSTR                            % CALL ABSTR, SAVE STATUS
124411               IF DMFLD.TYPRING BIT 5FLOP THEN
124415                  IF SVXERR BIT 4 THEN A:=0 ELSE FIMAXW FI
124423                  A=:IMAXW
124424               ELSE
124425                  IF A BIT 5IBDV THEN
124427                     MDFI.BUFFER; A=:MDBUF=:CMDBUF      % INTERNAL BLOCK DEVICE.
124433                     X.DBLOCK=:IMAXW
124435                  ELSE
124436                     IF A BIT 5HDMA THEN
124440                        IF SVXERR BIT 4 THEN
124443                           A:=0
124444                        ELSE
124445                           X:=MDFI.BUFFER; T:=BUFBANK; *LDATX 10
124451                           A+1 SHZ -1+2
124454                           IF A>IMAXW THEN
124457                              A:=T
124460                           FI
124460                        FI
124460                        A=:IMAXW
124461                     ELSE
124462                        MDFI.DBLOCK=:IMAXW
124465            FI FI FI
124465    RRETU:  X:=B; CALL GAPIT; X:="IRETW"; T:=IMAXW
124471            IF MTFLG BIT MCMAGTP THEN
124474               CALL DALTON
124475               T=:X.SO; CALL ALTOFF; *BSET ONE
124500            FI; GO L2                                   % NO. OF WORDS READ
124501    *)FILL
124512
124612    L2:     IF T>0 THEN
124514               OFLD; CALL XLOCK
124516               IF IFUNC=26 THEN
124522                  IMAXW+1 SHZ -1 =: IMAXW               % BYTES => WORDS!!!
124526               FI; OLDPAGE=:D
124530    @LIB CXCPU-,
124530               X:=CMDBUF; ICORAD; T:=BUFBANK; CALL DBTRANS;
124534               T:=IMAXW; CALL PT3COPYB                  % LAST BLOCK <= 2000 (OCT) WORDS.
124536               OFLD; CALL XUNLOCK
124540            FI
124540            GO OVER
124541    *)FILL
124546                                        % WRITE
124546    L3:     OFLD; CALL XLOCK
124550            IMAXW=:D
```

```
124552                IF IFUNC=27 THEN
124556                   A:=D+1 SHZ -1 =: D                          % WRITE BYTE RECORD.
124561                FI                                             % BYTES => WORDS!!!
124561    @LIB CXCPU-,
124561                T:=OLDPAGE:=:D; X:=CMDBUF; ICORAD
124565                T=:TSAVE; T:=BUFBANK; CALL DBTRANS; T:=TSAVE
124571                A:=:X; CALL PT3COPYB
124573                OFLD; CALL XUNLOCK
124575                CALL CALABSTR
124576    OVER:       CALL BUNLOCK; GO RETU1
124600
124600    INTEGER POINTER OFLD:=OFLCK
124601                                                   % IN FILE SYSTEM
124601    RETU1: IF DMFLD.OFFLG BIT 5RCFIL AND X.TYPRING BIT 5RFILE GO RETU3
124610           IF SVXERR BIT 4 THEN X:=DMFLD; CALL MERRCODE ELSE A:=0 FI
124617    RETU:  A=:SVXERR
124620    RETU3: IF DMFLD.TYPRING BIT 5IBDV AND IFUNC >< 21 THEN
124630              A:=MDFI.LNUMB; CALL UNLOC
124633           ELSE                                                % UNLOCK, BUT DON'T CLEAR, DBH.
124634              IF A NBIT 5HDMA THEN
124636                 OFLD; CALL XLOCK
124640                 T:=IOLOG; CALL R3BUF; CALL ERRFATAL
124643                 OFLD; CALL XUNLOCK; A:=SVXERR
124646              FI
124646           FI
124646    RETU2: X:=ZDREG=:D:=ZXREG; T:=ZTREG; GO MCLRG
124653
124653    *)FILL
124666
124666    % SUBROUTINE FOR CALLING ABSTRANS
124666    CALABSTR:
124666           IF DMFLD.OFFLG BIT 5RCFIL AND X.TYPRING BIT 5RFILE THEN      % TEST IF REMOTE
124675              B=:D; T:=MDFI;  GO RMAGT                                  % REMOTE MAGTP
124700           FI
124700           IFUNC=:MDFI.DKFUN; IUNIT=:X.DBLO1; IMAXW=:X.DBLOCK
124707           IF DMFLD.TYPRING BIT 5FLOP THEN
124713              IF IFUNC<2 OR A=43 OR A=44 THEN                           % READ/WRITE FUNCTIOINS
124725                 IUNIT+DMFLD+"FDIFORM"=:X; X.SO+DMFLD+"WDSCT"=:X
124735                 T:=X.SO; A:=IMAXW=:D:=0; *RDIV ST
124742                 IF D><0 THEN A+1 FI; A=:MDFI.DBLOCK=:D
124750                 IUNIT+DMFLD+"NFDIADR"=:X; X.SO+D=:X.SO-D=:MDFI.DBLO1
124762              FI
124762              IUNIT SH 6+IFUNC=:MDFI.DKFUN
124767           ELSE
124770              IF A BIT 5IBDV AND IFUNC/\ 77 = 1 THEN
124777                 DMLOG BONE 17=:DMLOG
125002              FI                                                        % WRITE ON AN INTERNAL BLOCK DEVICE.
125002           FI
125002           T:=DMLOG; MDFI+"DPNTO"; *MON 2ABST
125006           A=:SVXERR; EXIT
125010    *)FILL
125016
125016           *SDATA
125016    DISP 0
125016    INTEGER XREG,TREG,AREG,DREG,LREG,BREG
125016    INTEGER DBHAD,SAVSG,FUDSG,OLSGM,CFRSEG
125016    INTEGER ARRAY XBMSV(10)                                % SAVE ARRAY FOR BITMAP
125016    INTEGER WNUMB,ZFUNC,REPAR,WMOVD,FSPEC
125016    INTEGER MDBU1,MDBU2; DOUBLE XMDBUF=MDBU1
125016    INTEGER FUBU1,FUBU2; DOUBLE FUBUF=FUBU1
```

```
125016   PSID
125016
125016          *DATA FUBU2
125016
125016   % AT ENTRY;
125016   %
125016   % XREG: POINTER TO OPEN FILE TABLE
125016   % TREG: ADDRESS TO DEVICE BUFFER HEADER
125016   % DREG: ADDRESS TO "DF-DATAFIELD"
125016
125016   DISP 24; INTEGER OFADR,MGFUNC,RWFUN,DAADR,NWRDS,ANWRDS,RTPAR; PSID
125016   RMAGT:
125016   *"-8RFAC
"125016           *FENTR
125023          T=:DBHAD; T.DBPAG=:XMDBUF
125027          RTREF.RSEGM=:CFRSEG; X.ACTSEG=:SAVSG
125034          A:=DRFSG;  A\/177400=:FUDSG
125037          0=:FUBU1; TLBUF=:FUBU2; 0=:FSPEC
125043          % SAVE BITMAP FOR CALLER
125043   *"CXCPU
"125043           X.RTDLGADDR+5BITMAP=:D; A:=0; X:=10=:L; T:="XBMSV"+B; *MOVPA
125054   *"-CXCPU
"125054
125054          A:=DREG.IFUNC=:ZFUNC; T:=X.IMAXW=:WNUMB
125061   @ICR
125061          IF A>4 AND ><43 AND ><44 AND ><25 AND ><52 AND ><55
125061             AND ><26 AND ><27 THEN                                    ...
125061   @CR;
125111             4=:FSPEC; CALL SPARM; CALL PT3FUSER; GO FAR ERET
125116             IF ZFUNC=42 OR =46 OR =56 OR =63 THEN
125133                WMOVD=:DBHAD.DBLOCK;
125136             FI; REPAR=:DREG.SVXERR=:AREG; GO OUT3; *)FILL
125155          FI
125155          % READ OR WRITE FUNCTIONS:
125155          IF A=26 OR =27 THEN  % NUMBER OF BYTES IN IMAXW
125163             WNUMB+1 SHZ -1=:WNUMB; FSPEC+2=:FSPEC
125172          FI
125172          IF WNUMB<0 OR >2000 THEN A:=172; GO ERET; FI     % MAX 1 K
125201          IF ZFUNC><0 AND ><2 AND ><43 AND ><25 AND><26 AND ><36 GO MWRT
125222          % READ:
125222          CALL SPARM; CALL PT3FUSER; GO ERET;
125225          % MOVE DATA FROM FILE USER DATA SEGMENT TO DEVICE BUFFER:
125225          RTREF.ACTPRI=:D; T:=FUDSG; CALL MMEXV; T=:OLSGM; A:=0; CALL MMREENT
125235          X:=MDBU2; T:=MDBU1; A:=TLBUF; CALL DBTRANS; X:=:A
125242          T:=WNUMB; CALL PT3COPYB; T:=OLSGM; CALL MMEXV; A:=CFRSEG; CALL MMREENT
125250          WMOVD=:DREG.IMAXW; REPAR=:X.SVXERR; "RRETU"=:LREG; GO OUT3; *)FILL
125272   MWRT:  % WRITE:
125272          % MOVE DATA FROM DEVICE BUFFER TO FILE USER DATA SEGMENT:
125272          RTREF.ACTPRI=:D; T:=FUDSG; CALL MMEXV; T=:OLSGM; A:=0; CALL MMREENT
125302          X:=MDBU2; T:=MDBU1; A:=TLBUF; CALL DBTRANS;
125306          T:=WNUMB; CALL PT3COPYB; T:=OLSGM; CALL MMEXV; A:=CFRSEG; CALL MMREENT
125314          FSPEC+1=:FSPEC; CALL SPARM; CALL PT3FUSER; GO ERET; A:=REPAR
125323   ERET:  A=:AREG=:DREG.SVXERR; "RETU"=:LREG
125330   OUT3:  X:=RTREF; % RESTORE CALLERS BITMAP:
125331   *"CXCPU
"125331           T:=X.RTDLGADDR+5BITMAP; X:=0; A:="XBMSV"+B=:D:=10=:L; *MOVAP
125342   *"-CXCPU
"125342           *FLEAV
125344   *)FILL
125357
```

```
125357    SPARM: A:=XREG=:CSTCK.OFADR; "FUBU1"+B=:X.DAADR; "ZFUNC"+B=:X.MGFUNC
125370           "WNUMB"+B=:X.NWRDS; "WMOVD"+B=:X.ANWRDS; "REPAR"+B=:X.RTPAR
125401           "FSPEC"+B=:X.RWFUN; X:=UMAGTP; EXIT
125406    *"
"125406    RBUS
125415
125415    *"BRFAC
"125415    %
125415    % ============================================================================
125415    % 15.16       P T 3 F U S E R
125415    % INTERFACE ROUTINE TO CALL PLANC SUBROUTINES ON FILE USER SEGMENT
125415    % PARAMETER TRANSFER BY REFERENCE
125415    %
125415    % CALL SEQUENCE:
125415    %       X:=<INDEX TO PLANC ROUTINE>; CALL FILUS; CALL ERROR
125415    %       SKIP RETURN IF NO ERRORS DETECTED
125415    % NOTE:
125415    %       BEFORE CALLING FILUS, REFERENCES TO PARAMETERS MUST BE
125415    %       PLACED ON THE FILE SYSTEM STACK FROM LOCATION CSTCK+22
125415    %       CSTCK: FILE SYSTEM CURRENT STACK POINTER
125415    %
125415    SUBR PT3FUSER
125415
125415    INTEGER WPROG=?
125415    INTEGER ARRAY POINTER RFRES:=RFSTB       % RFA DATA SEGMENT RESERVE TABLE
125416    INTEGER POINTER NMSEGS:=NRFSG            % NUMBER OF RFA DATA SEGMENTS
125417    INTEGER POINTER CRTR:=CRTREF             % CALLING RT PROGRAM
125420
125420    DISP 0; DOUBLE DACTSEG=ACTSEG; PSID
125420
125420           *SDATA
125420
125420    % NOTE:       THIS DISP VARIABLES ARE ALSO DEFINED IN THE ROUTINE
125420    %             TOFENTRY IN RESIDENT (ON FILE SINB-X).
125420    %             DO NOT CHANGE THIS DISP DEFINITIONS WITHOUT DOING
125420    %             THE SAME MODIFICATIONS IN TOFENTRY
125420    %
125420
125420    DISP 0
125420           INTEGER XREG,TREG,AREG,DREG,LREG,BREG
125420           INTEGER OLDSG,OLDRSG                        % SAVED ON FILE SYSTEM STACK
125420           INTEGER RETUR,PREVB,STPTR,STMAX,LEXIT,ERRCO          % PLANC STACK FRAME
125420    PSID
125420           *DATA ERRCO
125420
125420    PT3FUSER: *FENTR
125425           0=:LEXIT=:ERRCO        % MAKE PLANC STACK FRAME
125427           "ESTCK"=:STMAX
125431           CSTCK=:STPTR; X:=RTREF
125434           IF BACKGR=0 OR X.ACTSEG SHZ -10=5ERRSEG OR XREG=URTLIOP THEN
125447              % RT PROGRAM OR RTOPEN, RTCONNECT, RTCLOSE,LIST-RTOPEN OR RTRES-O-F-E
125447              A:="5RRUS"=:DRFSG  % RT PROGRAM
125451           ELSE
125452              IF DRFSG=0 THEN
125454                 *IOF                                                    %%%%%%%%%%%%%%%%%%%%
125455                 FOR X:=0 TO NMSEGS DO                                   %%                  %%
125461                    IF RFRES(X)=0 GO SFNDX; X=:T:=A  % SEGMENT IS FREE TO USE   %%  MUST NOT CROSS %%
125465                    IF X.WLINK=0 AND X.TLINK=0 AND X.STATUS NBIT 5RWAIT THEN    %%  A PAGE BORDER  %%
125474                       T=:X; GO SFNDX   % PROGRAM IS PASSIVE, TAKE SEGMENT      %%                  %%
125476                    FI; T=:X                                               %%                  %%
```

```
125477                OD; *ION                                                          %%                    %%
125502                % ALL SEGMENTS ARE RESERVED                                      %%                    %%
125502                A:=ER188=:AREG; GO FAR NSGSX; *)FILL                             %%                    %%
125516    SFNDX:      RTREF=:RFRES(X); 0=:INSFG; A:="DSSNM"+X=:DRFSG; X:=RTREF; *ION  %%%%%%%%%%%%%%%%%%%%
125526             FI
125526          FI    % A-REG: RFA DATA SEGMENT NUMBER
125526          A\/177400=:T;
125530          CALL MMEXY; T=:OLDSG; X.RSEGM=:OLDRSG;
125534          IF X:=XREG<=UCLOS THEN A:=5FIU2 ELSE A:=5FIUS; FI; CALL MMREENT
125544          IF DRFSG="5RRUS" THEN           % RT PROGRAM
125550             IF X:="RTRFA"=RTREF GO ASBACK          % ALREADY RUNNING
125554             A:=1730; CALL XLOCK                    % RESERVE RTRFA SEMA
125556             X:=X.RTDLGADDR; CALL SVRBLK            % COPY REGISTER BLOCK
125560             A:="NSTRT"; T:=0; *STATX               % SET P-REG.
125563             AD:=RTREF.DACTSEG; D BZERO 5BACKGR; AD=:"RTRFA".DACTSEG
125570             RTREF.RSEGM=:"RTRFA".RSEGM
125574             % RESERVE DF DATAFIELD
125574             MLEV; *MCL PIE
125576             X:="RTRFA"; CALL RTENTRY; RTREF.STATUS BONE 5WAIT=:X.STATUS
125604             "STUPR"; *IRW MLEVB DP;
125606             MLEV; *MST PID; MST PIE
125611             GO XRRT; A:=1730; CALL XUNLOCK; MIN LREG; GO OUT
125616    XRRT:    A:=1730; CALL XUNLOCK; D=:A; GO ERRT
125622    NSTRT:   T=:WPROG                               % NEW ENTRY OF RTRFA
125623             X:=XREG; CALL TOFENTRY; GO EERRT
125626             T:=0; X:=WPROG.RTDLGADDR
125631             *LDATX; AAA 1; STATX                   % INCREMENT P-REG.
125634             GO ENDXX
125635    EERRT:   X:=WPROG.RTDLGADDR; T:=0; *STATX 40    % ERROR CODE IN D-REG.
125641    ENDXX:   WPROG.STATUS BZERO 5WAIT=:X.STATUS
125645             *JMP I (SRRT1; )FILL                   % RESTART WAITING TASK
125667             INTEGER WPROG
125670          FI
125670    ASBAC: X:=XREG
125671          CALL TOFENTRY; GO ERRT                    % X MUST BE SET
125673          MIN LREG; GO OUT                          % SKIP RETURN IF NO ERRORS
125675    ERRT:  IF A=-1 THEN A:=ER161; FI; A=:AREG
125702    OUT:   % GET BACK FILE SYSTEM SEGMENT(S);
125702          T:=OLDSG; CALL MMEXY; A:=OLDRSG; CALL MMREENT
125706    NSGSX: *FLEAV
125710    RBUS
125715    *"
"125715
125715    *"8C1X2+8C2X2+8C3X2+8C4X2
```

```
125715   %
125715   %=============================================
125715   % 40.20
125715   %       X 2 1 2 S     X 2 1 3 S     X 2 S N D
125715   %       X 2 S S S
125715
125715   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
125715   %
125715   %       X 2 1   DRIVER FOR HANDLING X21 DCB'S
125715   %
125715   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
125715
125715
125715   %PURPOSE:      CONNECT AND DISCONNECT TO A X 21 NETWORK
125715
125715   %ENTER:        B-REG DATAFIELDPOINTER
125715   %              MESSAGE:        COMMAND
125715   %                              FACILITY
125715   %                              STATUS
125715   %                              SELECTIONS SIGNALS
125715   %                              DTE/DCE PROVIDED INFORMATION
125715
125715
125715   %EXIT:         MESSAGE SENT BACK TO USER WITH APPROPRIATE STATUS INFORMATION.
125715
125715   %HDLC REGISTER CONTENT DEFINITION
125715   %
125715   %
125715   SYMBOL CPCR = 100
125715   SYMBOL CSAR = 377% IOX + 3                    % IOX + 1
125715   SYMBOL CCL  = 347
125715   SYMBOL CTCW =   1% IOX + 7                    %IOX + 4  7BITS BYTE+PARITY
125715   SYMBOL CWTTC= 104% IOX + 13
125715   SYMBOL CRTTC= 104% IOX + 11
125715   %
125715   %
125715
125715   %
125715   %
125715   %       X21 LOGGING ROUTINES
125715   %**************************
125715
125715
125715           SYMBOL X22SIZ = 20
125715           SYMBOL X2LMA = X22SIZ-1
125715           SYMBOL DCEMS = X22SIZ*10+4
125715
125715           INTEGER ARRAY X2A00(X22SIZ)
125735           INTEGER ARRAY X2A02(X22SIZ)
125755           INTEGER ARRAY X2A05(X22SIZ)
125775           INTEGER ARRAY X2A10(X22SIZ)
126015           INTEGER ARRAY X2A11(X22SIZ)
126035           INTEGER ARRAY X2A12(X22SIZ)
126055           INTEGER ARRAY X2AST(X22SIZ)
126075           INTEGER ARRAY X2ATL(X22SIZ)
126115
126115           INTEGER X2LL:=0
126116           INTEGER X2LL1:=-1
126117           INTEGER X2LL2:=0
```

```
126120          INTEGER X2LL4:=0
126121          INTEGER X2LL5:=0
126122
126122    %
126122    %
126122    %       LOGG IN ARRAYS
126122
126122    SUBR STORE
126122    STORE:  IF X2D00 >< X2A00(X2LL) THEN GO XUP21 FI
126130            IF X2D02 >< X2A02(X2LL) AND >< "0" THEN GO XUP21 FI
126141            IF X2D05 >< X2A05(X2LL) THEN GO XUP21 FI
126147            IF X2D10 >< X2A10(X2LL) THEN GO XUP21 FI
126155            IF X2D11 >< X2A11(X2LL) THEN GO XUP21 FI
126163            IF X2D12 >< X2A12(X2LL) THEN GO XUP21 FI
126171            IF X2DST >< X2AST(X2LL) THEN GO XUP21 FI
126177            MIN X2LL4; P+0; X:=X2DMP
126202            EXIT
126203    XUP21:  MIN X2LL1; P+0
126205            X2LL1 /\ X2LMA=:X2LL
126210            X2D00=:X2A00(X2LL)
126213            X2D02=:X2A02(X2LL); 0=:X2D02
126217            X2D05=:X2A05(X2LL)
126222            X2D10=:X2A10(X2LL)
126225            X2D11=:X2A11(X2LL)
126230            X2D12=:X2A12(X2LL)
126233            X2DST=:X2AST(X2LL)
126236            X2LL-1=:X2LL5
126241            IF X2LL5 < 0 THEN "X22SIZ-1"=:X2LL5 FI
126245            X2LL4=:X2ATL(X2LL5); 0=:X2LL4
126251            X:=X2DMP; EXIT
126253    RBUS
126270
```

```
126270
126270    %*****************************
126270    %
126270    %       SOME USEFUL ROUTINES
126270    %
126270    %*****************************
126270
126270
126270    %       SERVICE ROUTINE FOR UPATINNG IDENT TABLES
126270    %*************************************************
126270
126270
126270    %       EXCHANGE "X2OLD" IN IDENTTABLE WITH "X2NEW"
126270    %       "X2OLD" MAY ALSO BE FOUND IN EXTENDED IDENTTABLE
126270    %       ENTRY X212S - LEVEL 12(D)
126270    %       ENTRY X213S - LEVEL 13(D)
126270
126270    %       INPUT PARAMS:
126270
126270    INTEGER X2NEW:=0                                    % ADDR TO BE INSERTED
126271    INTEGER X2OLD:=0                                    % ADDR TO BE REMOVED
126272
126272    %       SCRATCH LOCATIONS
126272
126272    SUBR X212S,X213S
126272    INTEGER X2TA1:=0                                    % ADDR TO TABLE
126273    INTEGER X2TA2:=0                                    % ADDR TO EXTENDED TABLE
126274    INTEGER X2SSP:=0                                    % WORKING
126275
126275    DISP 0; INTEGER XNUL1; INTEGER EN; PSID
126275
126275    X212S: IDNTS(2)=:X2TA1+"MAX12"=:X2SSP; EXTDS(2)=:X2TA2; GO X2HER
126306    X213S: IDNTS(3)=:X2TA1+"MAX13"=:X2SSP; EXTDS(3)=:X2TA2
126316
126316    X2HER: FOR X:=X2TA1 STEP 1 TO X2SSP DO
126322               T:=0; *LDATX
126324               IF A=X2OLD THEN
126327                  T:=0; X2NEW; *STATX
126332                  EXIT
126333               FI
126333           OD
126335
126335    %       LOOK IN EXTENDED TABLE
126335
126335           DO T:=0; X:=X2TA2; X+1; * LDATX
126341           WHILE A><-1
126344               X:=X2TA2; X+1; T:=0; * LDATX
126350               IF A=X2OLD THEN
126353                  T:=0; X2NEW; *STATX
126356                  EXIT
126357               FI
126357               MIN X2TA2; P+0
126361           OD
126362           A:=X2FNI; CALL X21UT                       % ENTRY NOT FOUND
126364    RBUS
126372
126372    %
126372    %       SEND ONE CHARACTER
126372    %       THE ROUTINE WILL GENERATE ODD PARITY
126372    %
```

```
126372    %       INPUT: A-REG IS CHARACTER TO BE TRANSMITTED
126372    %
126372    %       OUTPUT: STATUS IN X2D12 IN DATAFIELD
126372    %
126372    %*****************************************************
126372
126372    SUBR X2SND
126372    X2SND:  T:=L=:X2DLS;
126374            CALL X21PG; A=:X2D05
126376            T:=X2DHD+XWTDR; *EXR ST
126401            105
126402            T+"XWTTC-XWTDR"
126403            *EXR ST
126404            CALL X21LV; T:=X2DHD+XRTTS; *EXR ST
126410            A=:X2D12; CALL STORE; A:=X2DLS=:P
126414
126414
126414    %       ROUTINE FOR PARITY GENERATION ( ODD PARITY)
126414    %*****************************************************
126414
126414    X21PG:  *BSET ONE 70 DA
126415            *BSKP ZRO 60 DA; BSET BCM 70 DA
126417            *BSKP ZRO 50 DA; BSET BCM 70 DA
126421            *BSKP ZRO 40 DA; BSET BCM 70 DA
126423            *BSKP ZRO 30 DA; BSET BCM 70 DA
126425            *BSKP ZRO 20 DA; BSET BCM 70 DA
126427            *BSKP ZRO 10 DA; BSET BCM 70 DA
126431            *BSKP ZRO  0 DA; BSET BCM 70 DA
126433            EXIT
126434    RBUS
126437
126437    %
126437    %       ROUTINE FOR STARTING TRANSMITTER PART OF INTERFACE
126437    %       AND SENDING "SYN,SYN"
126437    %
126437    %*************************************************************
126437
126437    SUBR X2SSS
126437    X2SSS:  A:=L=:X2DLS
126441            1=:X2DPC
126443            FOR X2DPC STEP 1 TO 4 DO                    % SEND 4 SYNC
126447                1; T:=X2DHD+XWTCR; *EXR ST
126453    L1;         105; T:=X2DHD+XWTTC; *EXR ST
126457                CALL X21LV
126460                T:=X2DHD+XRTTS; *EXR ST
126463                A=:X2D13
126464                IF A NBIT 0 THEN GO L1 FI
126467            OD
126473            0=:A; T:=X2DHD+XWTCR; *EXR ST
126477            A:=X2DLS=:P
126501    RBUS
126502
126502
```

```
126502
126502    % ============================================================
126502    % 40.21
126502    %       X 2 1 E R     X 2 S T A      X 2 S B Y     X 2 R B Y
126502    %       X 2 R A C     X 2 1 U T
126502    % ============================================================
126502
126502    %*******************************
126502    %
126502    %       MESSAGE HANDLING
126502    %
126502    %*******************************
126502
126502    SUBR X21ER,X2STA,X2SBY,X2RBY,X2RAC,X21UT
126502
126502    %
126502    %
126502    %       MESSAGE RECEPTION
126502    X2STA:  IF X2DST=10 THEN                                % ARE YOU IN READY STATE?
126506               CALL XTSTC; GO NOMES                         % YES, ANY MESSAGE?
126510                  CALL X2114                                % CONTROLLED NOT READY
126511                  0=:X2DST=:X2DPIN
126513                  X2FNC=:X2DMP.X2MST; CALL O3CHA      % YES, RETURN READY MESSAGE
126517                  0=:X2DMP; 1=:X2DDF
126522            FI
126522    NOMES:  CALL O3CHA; CALL X21LV; 0=:X2DER; CALL X2RBY       % GET NEW MESSAGE
126526            IF A >= 10 THEN CALL X21C FI                      % DECODE COMMAND
126532            IF A >= 6 THEN X2FSM=:X.CRET FI                   % TOO SMALL MESSAGE
126537            CALL O3CHA; 0=:X2DMP; GO X2STA                   % RETURN MINI MESSAGE
126542
126542
126542    %
126542    %
126542    %       MESSAGE SENDING        (RETURN IT TO SENDER)
126542
126542    X21ER:  A=:X2DMP.X2MST=:X2DER; CALL STORE; CALL X2116
126547            CALL X2121; CALL X211; CALL X2114
126552            X:=X2DMP; GO X21UU                                 % LEAVE AT STATE 24
126554
126554    X21UT:  A=:X2DMP.X2MST; CALL STORE                        % SET STATUS
126557    X21UU:   CALL X2RBY; A+X2DBC=:T; CALL X2SBY
126563            CALL O3CHA; 0=:X2DMP; 1=:X2DDF
126567            GO FAR X2STA
126570
126570    %
126570    %
126570    %       SOME USEFUL MESSAGE MAINTENANCE ROUTINES
126570
126570
126570    %       STORE BYTECOUNT IN MESSAGE
126570
126570    X2SBY:  T=:X.BBYTC; EXIT
126572
126572    %
126572    %       READ ACTUAL BYTECOUNT
126572
126572    X2RAC:  A:=X.BBYTC; EXIT
126574
126574    %
126574    %       READ MAX BYTECOUNT
```

```
126574
126574   X2RBY: A:=X.BMBYTE; EXIT
126576
126576   %
126576   %       ANY MESSAGES FOR ME?
126576
126576   XTSTC: IF IQUEU><0 THEN  EXITA FI
126601          EXIT
126602   RBUS
126620
126620   %=============================
126620   % 40.22
126620   %       X 2 1 C
126620   %
126620   %-=============================
126620
126620
126620   %*******************************
126620   %
126620   %       THE X21 COMMANDS
126620   %
126620   %*******************************
126620
126620
126620   SUBR X21C,X21RE,X21LG,X21CL,X21CN,X21CH,X21DC,X21AA,X21RD,X21BR .
126620
126620   %       S T A T U S
126620   %       ***********
126620
126620   X21AA: IF X2DCN = 0 THEN X21SA; GO LLL FI          % NOT CONNECTED
126624          IF X2DST = 14 THEN X21SB ELSE X21SC FI      % DATA-PHASE OR CONNECTED
126633   LLL:   T:=X2DBC+2=:X2DBC
126636          IF T:=X2DBC<0 THEN                          % ENOUGH MESSAGE SPACE ?
126641            A=:X2DMP.X2MSL; A:=0                      % YES
126644              ELSE
126645            A:=X2FSM                                  % NO
126646          FI
126646          CALL X21UT
126647
126647   %       C A L L
126647   %       *******
126647
126647   X21CA: IF X2DCN=0 THEN X2FFF; CALL X21UT FI        % MUST CONNECT FIRST
126653          IF X2DST=14 OR X2DCC = 2 THEN A:=X2FIS; CALL X21UT FI  % ILLEGAL COMMAND IN DATA PHASE
126665          IF X2DMP.X2MFA NBIT X21C3 THEN              % DIRECT CALL ?
126671             CALL X2RAC; A=:D; X:=0                   % NO, LOOK FOR "+"
126674             DO WHILE A><"X21PL"                      % ANY "+" IN SELECTION SIGNAL ?
126677                T:=X2DMP+X2M3; *LBYT
126702                X+1
126703                IF X>D OR X>15 THEN A:=X2FNP; CALL X21UT FI % NO, ERROR
126712             OD
126713          FI
126713          CALL X211
126714          IF T><0 THEN T=:A; CALL X21ER FI
126720          CALL X212                                   % STATE 2 - 6
126721   LOOP:  CALL X2GET; A=:A
126723             IF X2D10 BIT 6 THEN 0=:TMR; CALL X2112 FI  % DCE READY FOR DATA
126730             IF T = 0  GO LOOP                          % WAIT FOR DATA
126732             IF X2D00="X21SY" OR ="X21PL" OR ="177" THEN GO LOOP FI
126745             IF A="X21ST" OR ="X21SL" THEN
```

```
126754                  CALL X2110                                  % ENTER STATE 10
126754                  IF T><0 THEN T=:A; CALL X21ER FI
126760                     ELSE
126761                  0=:TMR; CALL X217; X21T3=:TMR               % CALL PROGRESS SIGNALS
126765                  IF T><0 THEN T=:A; CALL X21ER FI
126771               FI
126771            GO LOOP
126772      *)FILL
127003
127003      %        R E A D Y
127003      %        *********
127003
127003      X21RE:  IF X2DCN=0 THEN X2FFF; CALL X21UT FI            % MUST CONNECT FIRST
127007              IF X2DST=14 OR X2DCC = 2 THEN A:=X2FIS; CALL X21UT FI  % ILLEGAL COMMAND IN DATAPHASE
127021              CALL X211
127022              IF T><0 THEN T=:A; CALL X21ER FI
127026              CALL X218                                       % EXECUTE STATE 8 - 9
127027              DO CALL X2GET; A=:A                             %
127031              WHILE X2D10 NBIT 6
127034                 IF T=1 AND X2D00="X21ST" THEN CALL X2110       % CALLING LINE IDENTIFICATION
127044                   IF T><0 THEN 0=:TMR; T=:A; CALL X21ER FI  % TOO SMALL MESSAGE
127051              FI
127051              OD
127052              0=:TMR
127053              CALL X2112                                      % READY FOR DATA
127054
127054
127054      %        C L E A R  R E Q U E S T
127054      %        ************************
127054
127054      X21CL:  IF X2DCN=0 THEN X2FFF; CALL X21UT; FI           % MUST CONNECT FIRST
127060              IF X2DBR >< 0 THEN                              % BREAK REQUESTED
127062                X2FBR =: X2DBR.X2MST                          %
127065                0=:X2DBR; 4=:X.BBYTC                          % SET BYTECOUNT
127070                CALL O3CHA                                   % SEND IT TO USER
127071              FI
127071              CALL X2116; CALL X2121
127073              IF X2DCC=1 THEN                                 % EXPECTING CHARGING INFO ?
127077                CALL X21GC                                    % GET CHARGING
127100              FI
127100              CALL X21SH; CALL X2SSS; CALL X2114
127103              X21LO=:X2DUI.HXDOK=:X2DUO.HXDOK                 % LOCK USER DRIVER
127110              A:=0; CALL X21UT
127112      *)FILL
127130
127130      %        C O N N E C T  T O  X 2 1
127130      %        *************************
127130      *)FILL
127130
127130      X21CN:  IF X2DCN><0 THEN X2FAC; CALL X21UT FI           % ALLREADY CONNECTED
127134              CALL X2RAC
127135              IF A<16 THEN A:=X2FSM; CALL X21UT FI            % TOO SMALL MESSAGE
127142              A:=X2DMP.X2MSL; CALL LOGPH; A=:X2DUI            % GET USER INPUT DATAFIELD ADR
127146              IF A=0 THEN A:=X2FIL; CALL X21UT FI
127151              A:=X2DMP.X2MSS; CALL LOGPH; A:=D=:X2DUO         % GET USER OUTPUT DATAFIELD ADR
127156              IF A=0 THEN A:=X2FIL; CALL X21UT FI
127161              IF X2DMP.X2MS2=0 THEN A:=CURPR FI
127165              IF A><X2DUO.RTRES  THEN
127171                 X2FNR; CALL X21UT                            % NOT RESERVED BY YOU
127173              FI
```

```
127173          IF A><X2DUI.RTRES THEN
127177             X2FNR; CALL X21UT                        % NOT RESERVED BY YOU
127201          FI
127201
127201    %     GET NEW DATAFIELD IF HASP-DMA IS USED
127201
127201          IF X2DUO.TYPRING BIT 5HDMA THEN
127205            A:=X.CLOGDV; CALL LOGPH A=:X2DUO          % NB SHOULD HAVE BEEN OUTPUT
127210            A:=X2DUI.CLOGDV; CALL LOGPH; A=:X2DUI
127214          FI
127214
127214
127214    %     GET HARDWARE DEVICE NUMBER (HDEV)
127214
127214          A:=X2DUO.HDEV=:HDEV
127217          IF A><X2DUI.HDEV THEN X2FNM=:A; CALL X21UT FI  % NOT MATCHING HDEV
127226
127226    %     REROUTE INTERRUPTS TO X21 DRIVER
127226
127226          A:=B=:X2NEW; X2DUO=:X2OLD;
127232          CALL X212S                                   % UPDATE LEVEL 12
127233          A:=B=:X2NEW; X2DUI=:X2OLD
127237          CALL X213S                                   % UPDATE LEVEL 13
127240          1=:X2DCN; A:=0; CALL X21UT
127244    *)FILL
127254
127254    %     R E D I R E C T
127254    %     ***************
127254
127254    X21RD: A:=X2FIC; CALL X21UT
127256
127256    %     D I S C O N N E C T  F R O M  X 2 1
127256    %     ***********************************
127256
127256    X21DC: IF X2DCN=0 THEN A:=X2FFF; CALL X21UT; FI  % NOTHING CONNECTED
127262          IF X2DST=14 THEN A:=X2FIS; CALL X21UT; FI % YOU ARE IN DATA PHASE !
127270          A:=B=:X2OLD; X2DUO=:X2NEW; CALL X212S      % REROUTE LEVEL 12
127275          X2DUI=:X2NEW; CALL X213S                   % REROUTE LEVEL 13
127300          A:=0=:X2DCN; CALL X21UT                    % OK, RETURN
127303
127303
127303    %     G E T  C H A R G I N G  I N F O R M A T I O N
127303    %     *********************************************
127303
127303    X21CH: IF X2DCC >< 3 THEN X2FAB; CALL X21UT FI    % NO CHARGING RECEIVED
127311          IF X2DBC>-10 THEN X2FSM; CALL X21UT FI      % TOO SMALL MESSAGE
127317          0=:X2DPC
127320          DO WHILE X2DPC < 14
127324             T:=X2DCI; X:=X2DPC; *LBYT
127327             T:=X2DMP+X2M3; *SBYT
127332             MIN X2DPC; P+0
127334             MIN X2DBC; P+0
127336             IF A = "X21PL" THEN GO ASD FI
127342          OD
127343    ASD:  0=:X2DCC; A:=0; CALL X21UT
127346
127346    %     R E T U R N   W H E N   C O N N E C T I O N   B R O K E N
127346    %     *********************************************************
127346
127346    X21BR: IF X2DST >< 14 THEN X2FBR; CALL X21UT FI  % NOT IN DATAPHASE
```

```
127354          X =: X2DBR; 1=:X2DDF
127357          GO FAR X2STA
127360  %*******************************
127360  %
127360  %       X 2 1 COMMAND DECODER
127360  %
127360  %*******************************
127360
127360  X21C:   X=:X2DMP; A-; A+10=:X2DBC; A:=X.X2MFU-      % POSITIVE FUNCTIONS
127366          IF A < 1 OR > X21PM THEN
127374             A:=X2FIC; CALL X21UT                      % ILLEGAL COMMAND
127376          FI
127376          0=:X2DDF                                     % EXPECTING NO CHARGING INFO
127377          GOSW X21C,FAR X21CA,X21RE,X21CL,X21CH,X21CN,X21DC,X21RD,FAR X21AA,X21BR
127412  RBUS
```

```
127432
127432    %=========================================
127432    % 40.23
127432    %       X 2 1 L V     X 2 1 I N     X 2 1 G C
127432    %
127432    %=========================================
127432
127432    %*****************************************
127432    %
127432    %       INTERRUPT HANDLING ROUTINES
127432    %
127432    %*****************************************
127432
127432    SUBR X21LV,X21IN
127432
127432    %       LEAVE X21, WAITING FOR INTERRUPT
127432
127432            INTEGER CPOF(0); *POF
127433    X21LV:  A:=L=:X2DL3; *TRA STS
127436            IF A BIT 10 THEN GO FAR WT13 ELSE GO FAR WT12 FI
127442    %
127442    %
127442    %       LEVEL 12 AND 13 INTERRUPT HANDLING
127442
127442    X21IN:  *TRA STS                                        % WHICH LEVEL ?
127443            IF A BIT 10 THEN
127445
127445    %       LEVEL 13
127445    %       ********
127445
127445            T:=X2DHD+XRRTS; *EXR ST                         % GET STATUS
127450            A=:X2D10
127451            IF A BIT DCE19 THEN                             % DCE CLEAR INDICATION ?
127453             IF X2DDF >= 0 THEN                             % NOT ENTRY HERE TWICE
127455              A BONE 17 =: X2DDF
127457              IF X2DST >< 25 THEN                           % DON'T CARE IF WAITING FOR DCE TO GET READY *H*
127463                CALL X2120; 1=:WAKEF                        % YES
127466                X21LO=:X2DU1.HXDOK=:X2DU0.HXDOK             % LOCK USER DRIVER
127473                X21T6=:TMR; CALL X2121
127476                IF X2DCC = 1 THEN CALL X21GC FI             % GET CHARGING IF REQUESTED
127503                CALL X21SH; CALL X2SSS; CALL X2114          % NB X2SSS BRINGS US FROM LEVEL 13 TO LEVEL 12
127506                0=:WAKEF
127507                IF X2DBR >< 0 THEN                          % BREAK REQUESTED ?
127511                  X2FBR =: X2DBR.X2MST; 0=:X2DBR            % YES
127515                  4=:X.BBYTC; CALL O3CHA                    % SEND BREAK MESSAGE FOR USER
127520                FI
127520                IF X2DDF NBIT 0 THEN                        % X21 MODUS ?
127523                  X2DST+30=:X2DMP.X2MST; 6=:X.BBYTC         % YES, STATUS TO DCB
127531                  CALL O3CHA; 0=:X2DST=:X2DPI=:X2DMP        % SEND IT BACK TO USER
127535                  1=:X2DDF; GO FAR X2STA                    % NOT X21 MODUS ANY MORE
127540                FI
127540                X2DDF BONE 16; A BZERO 17=: X2DDF         %
127544               FI
127544             FI
127544            FI
127544
127544    %       ACTIVATE X21 DRIVER ?
127544
127544            IF X2DDF NBIT 0 OR A BIT 17 THEN              % X21 ACTIONS ?
127551              IF X2DPIN = 0 THEN GO X21LV FI              % NOT PINNED, SKIP IT
```

```
127554              A:=X2D10; 0=:X2DPIN; T:=X2DL3=:P          % GO TO X21 ROUTINE
127560            FI
127560
127560      %     ACTIVATE DATA DRIVER
127560
127560            IF X2DDF BIT 16 THEN                      % CONNECTION BROKEN ?
127563              A:=X2DUI=:B; T:="DRIVER"+3; A:=HX21M
127570              *IOF; IRW LV138 DA
127572              CALL S3L13; *ION                        % FIRST START DRIVER, THEN RETURN
127574              CALL WT12
127575                ELSE
127576              A:=X2D10                                %
127577              T:=X2DUI=:B; T:="DRIVER"+3; GO FAR X3T13 % GO TO USER DRIVER
127604            FI
127604            FI
127604
127604      %     LEVEL 12
127604      %     ********
127604
127604            IF X2DDF > 0 THEN                         % DATA PHASE ?
127607              A:=X2DUO=:B; GO FAR X3T12                  % YES, GO TO USER DRIVER
127612            FI
127612            A:=X2DL3=:P
127614                                                      % LEVEL = 12
127614      RBUS
127633
127633      SUBR X21GC
127633
127633      %     ROUTINE FOR RECEIVING CHARGING INFORMATION
127633      %     ******************************************
127633
127633      X21GC: A:=L=:X2DSL
127635            2=:X2DCC
127637              CALL X211; X21T7=:TMR; CALL X218        % WAITING FOR CHARGING
127643              0=:X2DPC=:WAKEF                         % YES, GET READY FOR IT
127645      KAST:   CALL X2GET; *JMP *-1
127647                IF X2D00 = "X21BL" OR A = "X21SY" THEN
127656                  GO KAST
127657                FI
127657                0=:TMR
127660                X:=X2DPC; T:=X2DCI; *SBYT           % STORE CHARGING INFO.
127663                MIN X2DPC; P+0                      % IN DATAFIELD
127665                IF X2D00 >< "X21PL" AND X2DPC < 14 THEN
127675                  GO KAST
127676                FI
127676              3=:X2DCC; CALL X2116; CALL X2121      % TERMINATE CHARGING CALL
127702              CALL X211
127703            A:=X2DSL=:P
127705      RBUS                                          % EXIT
```

```
127712
127712   %=============================
127712   % 40.24
127712   %       X 2 1 T O     X 2 G E T
127712   %
127712   %=============================
127712   %*************************
127712   %
127712   %       TIMOUT HANDLING
127712   %
127712   *************************
127713
127713   %       NON RESIDENT PART
127713
127713   SUBR X21TO
127713   X21TO: IF X2DPIN = 0 THEN GO FAR WT13 FI          % NOT PINNED, SKIP IT
127716          0=:X2DPIN
127717          IF X2DCC = 2 THEN 0=:X2DCC; CALL X2114; GO FAR WT13 FI          % WAITING FOR CHARGININFO ?
127726          IF X2DST = -1 THEN CALL X2114; X2F01; CALL X21UT; FI  %
127735          IF X2DST = 3 THEN CALL X21SH; CALL X2SSS; CALL X2114; A:=33; CALL X21UT FI  % STATE 1, STATE 14
127746          IF X2DST = 25 THEN CALL X2114              % HAVE SEEN DCE CLEAR, TIMOUT FOR 21
127753            IF X2DDF NBIT 0 THEN                       % X21 MODUS?
127756              55; CALL X21UT                           % 55 MEANS TIMOUT IN STATE 25
127760            ELSE
127761              IF X2DBR >< 0 THEN
127763                X2FBR =: X2DBR.X2MST; 0=:X2DBR
127767                4=:X.BBYTC; CALL O3CHA
127772              FI
127772              X21LO=:X2DUI.HXDOK=:X2DU0.HXDOK
127777              60000; T:=X2DUI=:B; T:="DRIVER"+3; GO FAR X3T13  % ACTIVATE DRIVER, 60000 MEANS CONNECTION BROK
130005            FI
130005          FI
130005          IF X2DER >< 0 THEN CALL X21SH; CALL X2SSS; X2F01; CALL X21UT; FI          % ERROR IN ERROR SEQUENCE
130013          X2DST+30; CALL X21ER
130016   RBUS
130027   %*****************************************************************************
130027   %
130027   %       WAIT FOR ANY INPUT REACTION
130027   %       THE INPUT CHARACTER IS STORED IN X2D00 IN DATAFIELD
130027   %       STATUS IS SAVED IN X2D10
130027   %
130027   %       RETURN: SKIP IF INPUT DATA ELSE NOSKIP
130027   %
130027   %*****************************************************************************
130027
130027   SUBR X2GET
130027   X2GET: A:=L=:X2DLS; -2=:X2D0C                      % SAVE RETURN ADDRES
130033          X2D11; T:=X2DHD+XWRTC; *EXR ST              % ENABLE RECEIVER, C = ON
130037          1=:X2DPI; CALL X21LV                        % PINNED FOR INPUT (SEE X21TO)
130042          A=:X2D10                                    % SAVE IT
130043          IF A BIT 1 THEN                             % STATUS AVAILABLE
130045                T:=X2DHD+XRRSR; *EXR ST               % GET STATUS
130050                A=:X2D02
130051          FI
130051          IF X2D10 BIT 0 THEN                         % ANY DATA AVAILABLE
130054             T:=X2DHD+XRRDR; *EXR ST                  % GET DATA
130057             A BZERO "7" =:X2D00; X2DLS+1=:X2DLS      % SKIP RETURN IF DATA
130064             T:=1                                     % T = 1  I.E DATA
130065          ELSE
130066             T:=0                                     % T = 0 I.E NO DATA
```

```
130067          FI
130067  OUT:    A:=X2DLS=:P                                     % NO DATA, NO SKIP
130071  RBUS
130072
130072
130072
130072  %=================================
130072  % 40.25       R E A D Y    X 2 1 S H    X 2 1 1 4
130072
130072  %*****************************************************************
130072  %
130072  %       R E A D Y                                       % ** STATE 1 **
130072  %       T = 1  C = OFF
130072  %
130072  %       EXIT: T = 0     OK
130072  %             T ><0     ERROR CODE IN T
130072  %
130072  %*****************************************************************
130072
130072  SUBR X211
130072  X211:   A:=L=:X2DSR; 1=:X2DST                           % SAVE RETURN ADDR
130076          CALL X21SH                                      % DEFINE SYN = HIGH, ENABLING
130077                                                          % INTERFACE TO DETECT R = HIGH
130077          100=:A; T:=X2DHD+XWTTC; *EXR ST                 % T = 1
130104          -3=:X2DOC; -1=:X2DST                            % **** POSSIBLE -7 HERE *****
130110  X21L:   005; T:=X2DHD+XWRTC; *EXR ST                    % C = OFF, IS R = 1 ?
130114          X21TA=:TMR; 1=:X2DPI; CALL X21LV                % WAIT FOR R = 1
130121          A=:X2D10                                        % SAVE STATUS
130122          IF A/\ "101" >< "001" THEN                      % IS I OFF ?
130126                T:=X2F01;  GO OUT                         % NO, REPORT ERROR
130130          FI                                              % YES,  I = OFF
130130          T:=X2DHD+XRRDR; *EXR ST                         % GET R
130133          A=:X2D00
130134          IF A >< "377" THEN                              % IS R = HIGH ?
130137                T:=X2F01; GO OUT                          % NO, NETWORK NOT READY
130141          FI                                              % I DO NOT LIKE STATE 8 NOW
130141          MIN X2DOC; GO X21L                              % MUST DETECT 3 SUCH CHAR, THUS
130143                                                          % MAKE SURE T = 1 IN 24 BIT
130143          0=:TMR                                          % INTERVALLS
130144          0=:A; T:=X2DHD+XWRTC; *EXR ST                   % DISABLE RECEIVER
130150          40; *EXR ST
130152          147; T:=X2DHD+XWPCR; *EXR ST                    % 8 BITS CHAR,STRIP SYNC
130156                                                          % NO PARITY CONTROL
130156          X21SY; T:=X2DHD+XWSAR; *EXR ST                  % REDEFINE SYNC CHARACTER
130162          100; T:=X2DHD+XWTTC; *EXR ST                    % T = 1
130166          CALL STORE; T:=0
130170  OUT:    A:=X2DSR=:P                                     % EXIT
130172  RBUS
130177
130177  %       SET READY (1,OFF) DONT CARE ABOUT DCE
130177
130177  SUBR X2118
130177
130177  X2118:  A:=L=:X2DSR; T:=22=:X2DST; CALL STORE
130204          CALL X21SH; 100; T:=X2DHD+XWTTC; *EXR ST
130211          A:=0; T+"XWRTC-XWTTC"; *EXR ST
130214          A:=X2DSR=:P
130216
130216  RBUS
130220
```

```
130220   %       STATE 14 CONTROLLED NOT READY
130220
130220   SUBR X2114
130220   X2114: A:=0; T:=X2DHD+XWRTC; *EXR ST
130224          A:=40; *EXR ST
130226          A:=0; *EXR ST
130230          A:=107; T+"XWPCR-XWRTC"; *EXR ST
130233          A:=252; T+"XWSAR-XWPCR"; *EXR ST
130236          A:=0; T+"XWCHL-XWSAR"; *EXR ST
130241          A:=104; T+"XWTTC-XWCHL"; *EXR ST
130244          A:=1; T+"XWTCR-XWTTC"; *EXR ST
130247          EXIT
130250   RBUS
130251   %
130251   %
130251   %       ENABLE INTERFACE TO SEE R = HIGH
130251
130251   SUBR    X21SH
130251   X21SH: A:=0; T:=X2DHD+XWRTC; *EXR ST             % DEVICE CLEAR
130255          A:=40; *EXR ST
130257          *AAT 6; EXR ST                           % CLEAR DMA ALSO *H*
130261          A:=107; T:=X2DHD+XWPCR; *EXR ST          % BCP MODUS, NO VCR CONTROL
130265                                                   % STRIP ONLY TWO SYN
130265          377; T:=X2DHD+XWSAR; *EXR ST             % SPECIFY SYNC CHAR
130271          A:=0; T:=X2DHD+XWCHL; *EXR ST            % CHARACTER LENGTH = 8 BITS
130275          EXIT
130276   RBUS
```

```
130277
130277    %============================================================
130277    % 40.26       X 2 1 1 9     X 2 1 2 0     X 2 1 2 1     X 2 1 1 6
130277
130277    %**************************************************
130277    %
130277    %       DTE/DCE CLEARING STATES  (16,17,19,20,21)
130277    %
130277    %**************************************************
130277
130277    $UBR X2119,X2120,X2121,X2116
130277
130277
130277
130277
130277    %       DCE CLEAR INDICATION                        ** STATE 19 **
130277    %       D = 0  I = OFF
130277
130277    X2119:  A:=L=:X2DSR; 23=:X2DST; CALL STORE
130304            A:=X2DSR=:P
130306
130306
130306
130306    %       CLEAR (DTE UNCONTROLLED)                    ** STATE 20 **
130306    %       T = 0  C = OFF
130306
130306    X2120:  A:=L=:X2DSR; 24=:X2DST; CALL STORE
130313            CALL X21SH                                   % SYN = HIGH TO DETECT STATE 21
130314            140; T:=X2DHD+XWTTC; *EXR ST                 % T = 0
130320            7=:X2D11; T+"XWRTC-XWTTC"; *EXR ST           % C = OFF
130324            A:=X2DSR=:P                                  % EXIT
130326
130326
130326    %       DCE READY (DTE UNCONTROLLED)                ** STATE 21 **
130326
130326    X2121:  A:=L=:X2DSR; 25=:X2DST; CALL STORE
130333            DO CALL X2GET; A=:A                          %
130335            WHILE T=0 OR X2D00><"X21EN" OR X2D10 BIT "6" OR X2DDF NBIT 17
130351                                                         % WAIT FOR R = 1, C = OFF,ST 21
130351            OD
130352            0=:TMR
130353            A:=X2DSR=:P                                  % EXIT
130355
130355
130355    %       DTE CLEAR REQUEST
130355    %       T = 0  C = OFF
130355
130355    X2116:  A:=L=:X2DSR; 20=:X2DST; CALL STORE           % RETURN ADDR
130362            CALL X21SH                                   % SYN = HIGH,
130363            140; T:=X2DHD+XWTTC; *EXR ST                 % T = 0
130367            207=:X2D11; X21T5=:TMR                       % C = OFF, TIMING UNTIL ST. 21
130373            A:=X2DSR=:P                                  % EXIT
130375    $BUS
130401
130401
130401
```

```
130401
130401    %====================
130401    % 40.27       X 2 1 2
130401
130401    %*************************************************************
130401    %
130401    %       DTE CALL REQUEST ROUTINES  (STATE 2,3,4,5)
130401    %
130401    %*************************************************************
130401
130401    SUBR X212
130401    %
130401    %       DTE CALL REQUEST                              ** STATE 2 **
130401    %       T = 0  C = ON
130401
130401    %
130401    X212:   A:=L=:X2DSR; 2=:X2DST; CALL STORE
130406            X21T1=:TMR; 307=:X2D11                         % C = ON
130412            140; T:=X2DHD+XWTTC; *EXR ST                   % T = 0
130416
130416    %       PROCEED TO SELECT ( WAIT FOR "+" )
130416
130416            3=:X2DST
130420            DO
130420               CALL X2GET; *JMP *-1                        % WAIT FOR DATA
130422            WHILE X2D00><"X21PL"  OD                       % WAIT FOR "+"
130427            0=:TMR                                         % RECEIVED "+"
130430
130430
130430    %       SEND SELECTOR SIGNALS                         ** STATE 4 **
130430    %       T = IA5  C = ON
130430
130430            300; T:=X2DHD+XWRTC; *EXR ST                   % MUST SEE DCE CLEAR REQUEST
130434            X21TB=:TMR; 4=:X2DST; CALL STORE
130441            IF X2DMP.X2MFA NBIT X21C3 THEN                 % SKIP ALL THIS IF DIRECT CALL
130445            CALL X2SSS                                     % SEND "SYN,SYN"
130446            IF X2DMP.X2MFA /\ 7 >< 0 THEN                  % ANY PREFIX ?
130452               IF A BIT X21C1 THEN                         % CHARGING REQUEST ?
130454                  ##6; CALL X2SND                          % YES, PREFIX = "61"
130456                  ##1; CALL X2SND
130460               FI
130460               IF X2DMP.X2MFA BIT X21C2 THEN               % CALLED LINE ID ?
130464                  IF A BIT X21C1 THEN
130466                     ##,; CALL X2SND
130470                  FI
130470                  ##6; CALL X2SND                          % YES, SEND ",62"
130472                  ##2; CALL X2SND
130474               FI
130474    %          IF X2DMP.X2MFA BIT X21C4 THEN               % THE USER MAY DO THIS
130474    %             IF A /\ 3 >< 0 THEN                      % EXPLISIT BY PREFIXING
130474    %                ##,; CALL X2SND                       % WITH "."
130474    %             FI
130474    %             ##.; CALL X2SND
130474    %          FI
130474               ##-; CALL X2SND                             % TERMINATION OF PREFIX
130476            FI
130476            0=:X2DPC
130477               DO                                          % NO, SEND SELECTION SIGNALS
130477                  X:=X2DPC; T:=X2DMP+X2M3; *LBYT
130503                  CALL X2SND
```

```
130504                  MIN X2DPC
130505                WHILE X2D05><"X21PP" OD                 % LAST CHAR IS "+"
130512    %           FI
130512                FI
130512                0=:TMR
130513                X:=X2DMP
130514                300=:A; T:=X2DHD+XWRTC; *EXR ST           % GET RID OF OVERRUN SITUATION
130521                T:=X2DHD+XRRDR; *EXR ST
130524                A=:X2D00
130525                T:=X2DHD+XRRTS; *EXR ST
130530                A:=X2D10; T:=X2DHD+XRRSR; *EXR ST
130534                A=:X2D02; CALL STORE
130536
130536
130536    %           DTE WAITING                             ** STATE 5 **
130536    %           T = 1  C = ON
130536
130536                5=:X2DST; 0=:A; T:=X2DHD+XWTTC; *EXR ST   % T = 1
130544                X21T2=:TMR; A:=X2DSR=:P                   % EXIT
130550    RBUS
```

```
130560
130560
130560   %=============================================================
130560   % 40.28       X 2 1 7     X 2 1 1 0     X 2 1 1 2     X 2 1 R T
130560   %*********************************************
130560   %
130560   %       INFORMATION FROM DCE
130560   %
130560   %*********************************************
130560
130560   %
130560   %
130560   %      CALL PROGRESS SIGNALS                          ** STATE 7 **
130560   %       T-REG AT EXIT:
130560   %       T = CODE GROUP NO
130560   %       T < 0 MEANS  T = ERRORCODE-
130560
130560   SUBR X217
130560   X217:   T:=7=:X2DST; A:=L=:X2DSR; CALL STORE       % ENTERING STATE 7
130565          0=:X2DPC=:X2DPS
130567          DO WHILE X2D00><"X21PL"
130573             IF T:=X2DBC<0 THEN
130576                X:=X2DPC
130577                T:=X2DMP+X2M2; *SBYT                   % SAVE BYTE IN MESSAGE
130602             ELSE
130603                T:=X2FSM; GO OUT                       % MESSAGE TOO SMALL
130605             FI
130605             MIN X2DPC; P+0
130607             MIN X2DBC; P+0
130611             IF X2D00><"X21PL" OR ><"X21KO" THEN
130620                X2DPS*12=:T; X2D00-60\/T=:X2DPS         % CONVERT FROM ASCII
130627             FI
130627             IF X2D00="X21KO" AND X2DPS<=24 THEN      % IMPORTENT CP SIGNAL?
130637                0=:X2DPS=:X2DPC                        % NO, FOREGET IT
130641             FI
130641             CALL X2GET; *JMP *-1                     % GET NEXT CHARACTER
130643          OD
130644          IF X2DPS>=24 THEN T:=X2FCP; GO OUT FI       % SERIOUS CP  STOPP
130652          IF X2DPC > 2 THEN T:=X2FEC; GO OUT FI       % EXTENDED CP SIGNAL
130660          T:=0
130661   OUT:   X:=X2DMP; A:=X2DSR=:P
130664   RBUS
130667
130667
130667   %       TAKE ACTION ACCORDING TO CALL PROGRESS SIGNAL
130667
130667
130667
130667   %
130667   %
130667   %       CALLED/CALLING LINE IDENTIFICATION          ** STATE 10
130667   %       OR DCE PROVIDED INFORMATION
130667
130667   SUBR X2110
130667   X2110:  T:=12=:X2DST;=L=:X2DSR; CALL STORE          % ENTERING STATE 10
130674          X:=0=:X2DPC
130676             DO WHILE X2D00><"X21PL"
130702                X:=X2DPC
130703                T:=X2DMP+X2M3; *SBYT                   % STORE IN MESSAGE
130706                MIN X2DBC; P+0                         % MESSAGE LENGTH INCREMENTED
130710                MIN X2DPC; P+0
```

```
130712                  IF X2DBC = 0 THEN T:=X2FSM; GO OUT FI % NO MORE SPACE IN MESSAGE
130716                  CALL X2GET; *JMP *-1
130720               OD
130721               X:=X2DPC; T:=X2DMP+X2M3; A:="X21PL"; *SBYT
130726               T:=0
130727     [illegible]:  A:=X2DSR=:P                                    % EXIT
130731     RBUS
130733
130733
130733
130733     %************************************
130733     %
130733     %       READY FOR DATA, AND LEAVE X21
130733     %
130733     %************************************
130733
130733     SUBR X2112,X21RT
130733
130733     %
130733     %
130733     %       READY FOR DATA
130733
130733     X2112: 0=:TMR; 14=:X2DST                                     % POSSIBLE DELAY HERE ?????
130736            304; T:=X2DHD+XWRTC; *EXR ST                          % TAKE CARE OF MODEM STATUS
130742            CALL STORE
130743
130743     %
130743     %
130743     %       LEAVE X21
130743
130743     X21RT: IF X2DMP.X2MFA BIT X21C1 THEN 1=:X2DCC FI % MARK IF CHARGING REQUEST
130751            IF X2DCC = 2 THEN 0=:X2DCC FI                         % WILL NOT SAVE OLD CHARGING
130756            X21OP=:X2DUI.HXDOK=:X2DUO.HXDOK                         % OPEN LOCK IN USER DRIVER
130763            X2FOK; CALL X21UT
130765
130765     RBUS
130770
```

```
130770
130770   %=========================================
130770   % 40.29      X 2 1 8         X 2 1 9
130770
130770   %*********************************
130770   %
130770   %       CALLED DTE
130770   %
130770   %*********************************
130770
130770   SUBR X218,X219
130770   %       INCOMMING CALL                                  ** STATE 8 **
130770
130770   X218:   10=:X2DST; A:=L=:X2DSR; CALL STORE
130775           CALL XTSTC; GO NOMES; CALL X2STA                % ANY NEW MESSAGES ON MY WAY IN HERE ?
131000   NOMES:  0=:WAKEF                                        % WAKE ME IF ANY MESSAGES
131001           7=:X2D11                                        % C= OFF
131003           DO WHILE X2D00><"X21BL"                         % WAIT FOR BELL
131007              CALL X2GET; *JMP *-1
131011           OD
131012
131012
131012   %
131012   %       CALL ACCEPTED
131012   %       T = 1, C = ON, R = BEL, I = OFF
131012   %
131012   X219:   1=:WAKEF; 307=:X2D11; 11=:X2DST                 % C = ON, ENABLE RECEIVER
131020           X21T4=:TMR; CALL STORE
131023           X2DSR=:P
131025   RBUS
131032   *"
"131032
131032   *"8C1X2
"131032   INTEGER ARRAY X2S01(200)
131232   INTEGER ARRAY X2E01(0)
131232   *"
"131232   *"8C2X2
"131232   *"8C3X2
"131232   *"8C4X2
"131232   *"8C5X2
"131232   *"8C6X2
"131232
131232   *"8UDMA+8PUDM+8VICO+8PVIC
"131232   SUBR PT3UDMA
131232   %-----------------------------------------------------------------
131232   % LEVEL 1  PON, ION  RING 2
131232   % ENTRY: X = WORKING FIELD  (IN OUTPUT DATAFIELD)
131232   %        B = PARAMETER LIST
131232   %---- -----------------------------------------------------------
131232     DISP  4; INTEGER OBSTATE; PSID       %
131232     DISP 20; INTEGER UFUNC; PSID         % FUNCTION CODE
131232     DISP 21; INTEGER UCORA; PSID         % USER MEMORY ADDRESS
131232     DISP 22; INTEGER UPA1H;
131232              INTEGER UPA1L;
131232        DOUBLE  UPAR1=UPA1H; PSID         % INPUT PARAMETER (LENGTH)
131232     DISP 24; INTEGER UOUTD; PSID         % OUTPUT DATAFIELD HAS ABSTR PARAMETER LIST
131232     DISP 25; INTEGER UPIOD; PSID         % PIO OUTPUT DATA
131232   % DISP 26; POINTER IRETW; PSID         % RETURN PARAMETER
131232   % DISP 27; INTEGER MTFLG; PSID
131232     DISP 30; INTEGER UMAXW; PSID         % LOOP COUNTER WHEN COPYING DATA TO/FROM USER
```

```
131232      DISP 31; INTEGER UBANK;              % MOST SIGNIFIKANT BUFFER ADDRESS
131232               INTEGER UBADR;              % LEAST SIGNIFIKANT BUFFER ADDRESS
131232           DOUBLE UBUFF=UBANK; PSID        % FULL BUFFER ADDRESS
131232      DISP 33; INTEGER ULOGD; PSID         % LOGICAL DEVICE
131232    % DISP 34; POINTER MCLRG; PSID
131232      DISP 35; INTEGER UFLAG; PSID         % USER FUNCTION FLAG
131232        SYMBOL BTUS=0              % COPY BUFFER TO USER
131232        SYMBOL BFUS=1              % COPY BUFFER FROM USER
131232        SYMBOL RETU=2              % RETURN PARAMETER TO USER
131232        SYMBOL DIRT=3              % DIRECT TRANSFER TO/FROM RT-PROG
131232        SYMBOL NRER=4              % NOT RETURN ERROR 171 (FOR READ STATUS)
131232        SYMBOL PIOI=5              % PIO IN PIO DATA IN 1 WORD BUFFER
131232        SYMBOL PIOO=6              % PIO OUT PIO DATA IN 1 WORD BUFFER
131232    DISP  0; DOUBLE POINTER DP2=P2; PSID
131232    DISP 7                        % ABSTRANS PARAMETER LIST IN OUTPUT DATAFIELD
131232        INTEGER POINTER FUNCP              % / *+4
131232        DOUBLE  POINTER MEMOP              % / *+4
131232        INTEGER POINTER BLOCP              % / *+5
131232        DOUBLE  POINTER DATP               % / *+5
131232
131232    PSID
131232    INTEGER POINTER OFLD=?
131232    PT3UDMA:
131232           0=:X.MTFLG
131233           CALL GAPIT; CALL DALTON;
131235           DP2; T:=P1;                  *BSET ZRO            % UPAR1 IS DOUBLE
131240           AD=:X.UPAR1; T=:X.UPIOD;     *BSET ONE
131243           T:="P1"; "P3";               *BSET ZRO
131246           A=:X."IRETW";                *BSET ONE
131250           CALL GET1;  T=:UCORA;  % GET PARAMETERS B = WORKING AREA
131252           ZTREG=:ULOGD
131254
131254           CALL LOGPH; IF A= 0 GO FAR EE033;
131257           D=:T; T=:UOUTD;     % SAVE OUTPUT DATAFIELD FOR ACCESSING ABSTR PARAMETER LIST
131261           IF A.TYPRING NBIT M144B GO FAR EE240          % UDMA LEGAL ON THIS DEVICE ?
131265
131265            IF BACKGROUND >< 0 THEN                    % BACKGROUND
131267               TTIFIEL.BSTATE=:OBSTATE=:COBSTATE
131273               X:=B+5REG;      *LRB BLEVB
131276               X:="ESCBLOCK"; *SRB BLEVB
131300               A:=B=:CMDFFIELD
131302           FI
131302           MLEV; *MST PIE;
131304
131304           "0"=:D
131306           IF UFUNC=0 THEN
131310              D BONE BTUS; D BONE RETU
131312             ELSE IF A = 1 THEN
131316                   D BONE BFUS
131317               ELSE IF A = 2  OR =  3 THEN
131326                    D BONE DIRT
131327                 ELSE IF A = 20 OR = 24 THEN
131336                      D BONE RETU; D BONE NRER
131340                   ELSE IF A = 54 OR= 56 THEN
131347                        D BONE PIOI
131350                     ELSE IF A = 55 OR= 57 THEN
131357                          D BONE PIOO;
131360                       ELSE IF A = 7 OR = 62 THEN
131367                            D BONE RETU;
131370                         ELSE IF A=21 OR=64 OR=65 OR=70 THEN % FUNCTION OK
```

```
131405                          ELSE GO FAR EE174; *)FILL
131422          FI FI FI FI FI FI FI FI
131422
131422          A:=D=:UFLAG
131424          T:=RTREF=:D
131426          IF A BIT BTUS OR BIT BFUS OR BIT PIOI THEN          % CHECK USER ADDRESS SPACE
131434             IF A BIT PIOI THEN
131436                T:=1
131437             ELSE
131440                T:=UPA1L;
131441             FI
131441             A:=UCORA;  CALL CHLIM; GO FAR EE153;
131444          FI
131444          IF UFLAG BIT RETU THEN
131447             A:="IRETW"; T:=2; CALL CHLIM; GO FAR EE153;
131453          FI
131453
131453          IF UFLAG BIT BTUS OR BIT BFUS OR BIT DIRT THEN
131462              OFLD; CALL XLOCK
131464              IF BACKGROUND >< 0 GO GBUF                    % RT PROGRAM
131466                 T:=RTREF.ACTSEG; X:=OLDPAG; AD:=UPAR1
131472                 UCORA; CALL XCSGM; GO GBUF;                % CHECK IF SEGMENT IS FIXC'ED
131475                 T:=UFLAG BZERO BTUS; T BZERO BFUS=:UFLAG % NO BUFFER TO/FROM USER
131501                 GO SBUFF; *)FILL
131510    GBUF:        IF UFLAG BIT DIRT GO EE241
131513               UPAR1;IF A >< 0 GO EE174; A:=D=:UMAXW        % DEVICE BUFFER << 64K
131517               T:=ULOGD; CALL G3BUFF; GO EE131
131522    SBUFF:     X:=UOUTD; X:=:B; AD=:MEMOP; B:=X; AD=:UBUFF
131527               OFLD; CALL XUNLOCK;
131531          FI
131531          IF UFLAG BIT BFUS THEN CALL DTFU FI           % DATA FROM USER TO BUFFER
131535
131535          IF UFLAG BIT PIOO OR A BIT PIOI THEN
131542            A:=UPIOD=:D; A:=UPA1L; AD=:UPAR1             % UCLIN BIT MASK + PIO DATA
131546          FI
131546          X:=UOUTD; X:=:B
131550          X.UFUNC=:FUNCP; X.UPAR1=:DATP; B:=:X
131555          T:=ULOGD;   A:=X; * AAA FUNCP; MON 2ABST        %  ABSTRANS
131561
131561          IF A < 0 AND UFLAG NBIT NRER THEN "171"  % RETURN ERRORCODE
131566          ELSE "0" FI
131570          A=:ZAREG
131571
131571          IF UFLAG BIT RETU OR A BIT PIOI THEN         % RETURN PARAMETER OR DATA
131576            X:=B; CALL GAPIT; CALL DALTON; *BSET ZRO
131602            X:=UOUTD; X:=:B; DATP; B:=X
131606            IF T:=UFLAG BIT PIOI THEN                  % RETURN ONE WORD DATA
131611               X:=UCORA; A:=D;        *BSET ONE
131614               A=:X.S0;
131615            ELSE                                       % RETURN DOUBLE WORD PARAM
131616               X:="IRETW";            *BSET ONE
131620               AD=:X.DS0;
131621            FI
131621            CALL ALTOFF;
131622          FI
131622          IF UFLAG BIT BTUS THEN  CALL DTTU FI         % COPY DATA TO USER BUFFER
131626
131626          IF UFLAG BIT BFUS OR BIT BTUS THEN
131633            OFLD; CALL XLOCK
131635            T:=ULOGD; CALL R3BUF; CALL ERRFATAL % RELEASE BUFFER
```

```
131640            OFLD; CALL XUNLOCK
131642          FI
131642          GO FINE
131643    INTEGER POINTER OFLD:=OFLCK
131644    EE033: "033"; GO ERR                                  % NO SUCH LOGICAL UNIT
131646    EE131:  OFLD; CALL XUNLOCK; "131"; GO ERR             % NO MORE BUFFER SPACE
131652    EE153: "153"; GO ERR                                  % ILLEGAL ADDRESS REF.
131654    EE174: "174"; GO ERR                                  % ILLEGAL PARAMETER
131656    EE240: "240"; GO ERR                                  % ILLEGAL DEVICE TYPE
131660    EE241:  OFLD; CALL XUNLOCK; "241";                    % SEGMENT NOT CONTIGUOSLY FIXED
131663    ERR:    A=:ZAREG;
131664    FINE:   IF BACKGROUND >< 0 THEN OBSTATE=:TTIFIEL.BSTATE; FI
131671          GO RET
131672    *)FILL
131710    %-------------------------------------------------------------------
131710    % LOCAL SUBROUTINE FOR COPYING DATA TO/FROM DEVICE BUFFER FROM/TO USER ARRY
131710    %-------------------------------------------------------------------
131710    DTTU:   K:="0"; GO DTF
131712    DTFU:   K:=1
131713    DTF:    A:=L=:"MCLRG"; OLDPAGE=:D;      OFLD; CALL XLOCK;
131721    @IIB CXCPU-,
131721          T:=UBANK; X:=UBADR; UCORA; CALL DBTRANS;
131725          T:=UMAXW; * BSKP IF ZRO SSK; SWAP SX DA
131730          CALL PT3COPYB; OFLD; CALL XUNLOCK;
131733          GO MCLRG;                % EXIT
131734    RBUS
131740    *"8F5UD
"131740    %=============================================================================
131740    %             U D R x x
131740    %
131740    % RT-PROGRAMS FOR FAST MON UDMA FROM ND-500
131740    %
131740    @ICR
131740    SUBR URT01,URT02,URT02,URT04,URT05,URT06,URT07,URT08,
131740         URT09,URT10,URT11,URT12,URT13,URT14,URT15,URT16;
131740    @CR;
131740    *"8UD01+8VI01
"131740    URT01: "0";GO FELL
131742    *"8UD02+8VI02
"131742
131742    DISP /                          % SPECIAL DF. FOR UDRXX RT-PROG
131742          INTEGER UR5M                              % ADDRESS TO MESSAGE FROM N500 PROC.
131742          INTEGER POINTER FUNCP                     % FUNCTION
131742          DOUBLE  POINTER MEMOP                     % MEMORY ADDRESS
131742          INTEGER POINTER BLOCP                     % NOT USED
131742          DOUBLE  POINTER NWORP                     % NR OF 16 BITS WORDS
131742    PSID
131742    SYMBOL H500A=7                  % CACHE CONTROLL IN MESSAGE
131742    SYMBOL SAP2=102                 % PAR2 VALUE (DATA ARRAY)
131742    SYMBOL SAP5=110                 % IPAR1 VALUE (RETURN PAR)
131742    SYMBOL XN5ST=20                 % ND-500 PROC. STATUS
131742    SYMBOL N5UDW=20                 % ND-500 PROC. STATUS = WAITING FOR FAST UDMA
131742    SYMBOL MBITMASK=12
131742          INTEGER HST,SVT
131744          INTEGER SVB=?
131744    FELL:   *IOF
131745          A+LUDV=:SVT                                %LOG DEV NO
131747          CALL LOGPH; IF A = 0 THEN CALL ERRF;  FI
131752          D=:A;       IF A = 0 THEN CALL ERRF;  FI
131755          A:=A.N5RDF=:B; X:=RTREF
```

```
131761          CALL BRESERVE; IF A < 0 THEN CALL ERRF;  FI
131764          DO %-----------------------------------  FOREVER
131764             T:=SVT;
131765             RTREF.STATUS BONE 5WAIT =:X.STATUS    % I/O WAIT UNTIL
131771            "RWAIT";* IRW MLEVB DP              % ACTIVATED FROM ND-500-DRIVER
131773             MLEV  ;* MST PIE; MST PID; ION
131777             A:=B;  * AAA FUNCP; MON 2ABST; IOF     % ABST TO ACTIVATE DRIVER
132003             A=:HST; T=:SVT
132005             X:=UR5M;T:=5MBBANK; * LDATX XN5ST
132010             IF A = N5UDW THEN                     % ND-500 PROC WAITING FOR UDMA
132013                IF HST < 0 AND FUNCP >< 20 AND A>< 24 THEN % READ STATUS
132024                  "171"; T:=1 ; ELSE "0"; T:=0
132031                FI
132031                AD SHZ -20; CALL TOPOF(MONICO)     % RETURN STATUS TO ND-500 PROC.
132034                IF FUNCP=0 OR =3 THEN              % CLEAR CACHE ON INPUT
132041                  T:=5MBBANK; X+H500A;*LDATX
132044                  A \/ 7400;          *STATX
132046                  X-H500A
132047                FI
132047                IF FUNCP=0 OR A=7 OR A=20 OR A=24 OR A=62 THEN
132065                  IF A=0 THEN AD:=NWORP SHZ 1      % WORDS TO BYTE
132070                  ELSE        AD:=NWORP
132072                  FI
132072                  T:=5AP5                          % T= DISP FOR IPAR2
132073                ELSE IF A >=54 AND A <=57 THEN     % PIO DATA
132102                  AD:=NWORP;T:=5AP2; "0"=:A        % T= DISP FOR MARY (DATA ARRY)
132106                ELSE
132107                  GO NRET; *)FILL                  % NO RETURN PARAMETER
132121                FI;FI
132121                X+T; T:=5MBBANK; * STDTX          % STORE RETURN PAR
132124                X:=UR5M+MBITMASK;* LDATX
132127                A BONE 4;         * STATX         % SET RETUN PARAMETER
132131                X-MBITMASK;
132132   NRET:        A:=:B=:SVB
132134                CALL TOPOF(XSETCPU)              % B= CPU DATAFIELD
132136                CALL TOPOF(TER500); A /\ A       % STOPP ND-500
132141                CALL TOPOF(LOWACT500); SVB=:B    % START ND-500
132145             FI
132145             *IOF
132146          OD
132147          INTEGER SVB
132150   ERRF:  *ION; MON 0
132152   RBUS
132154   *"
"132154
132154   *"8DILG
"132154
132154   %=============================================================================
132154   %              D I L R T
132154   %
132154   % RT PROGRAM FOR DISC-LOGGING ON FILE DISC-LOGG:LOGG
132154   %
132154
132154   SUBR DILRT
132154
132154
132154   INTEGER PMFIX:="DILGS"
132155
132155   DILRT: "DFDIL"=:B;
132157          DILGFLAG BZERO DILBOK=:DILGFLAG
```

```
132162          "PMFIX"; *MON 2UNFI
132164          "PMFIX"; *MON 2FIX
132166          DILGFLAG BONE DILBOK=:DILGFLAG
132171          DILFUNIT SH 6+61=:DALFUNC
132175          "DILGS"*5SEGSIZE+SEGSTART; X:=A.BPAGLINK; T:=CORMBANK; *LDATX DPAGP
132204          A=:D:=0; AD SH 12=:DDIBADDR; A:=D=:DILBPNT
132212          DO
132212  LOOP:       IF T:=DILGFLAG NBIT DILSTART GO OUT
132215              IF T BIT 1DILBFULL OR T BIT 2DILBFULL THEN
132221                  AD:=DDIBADDR
132222                  IF T BIT 2DILBFULL THEN T:=1000; D+T FI
132226                  AD=:DALCMADDR; DILDADDR=:DALCDADDR
132231                  T:=DILFLOG; "DALM131"; *MON 2ABST
132234                  IF A<0 THEN
132235                      *MON 64
132236                      GO OUT
132237                  FI; AD:=DILLADDR; A=:T; D=:L
132242                  AD:=DILDADDR; IF A=T AND D=L GO OUT
132247                  D+1; A:=A+C; AD=:DILDADDR
132252              FI;
132252              IF A:= DILGFLAG BIT 2DILBFULL THEN
132255                      A BZERO 2DILBFULL=:DILGFLAG
132257              ELSE
132260                      A BZERO 1DILBFULL=:DILGFLAG
132262              FI
132262              *IOF
132263              RTREF.STATUS BONE 5WAIT=:X.STATUS
132267              "RWAIT"; *IRW MLEVB DP
132271              MLEV; *MST PID; ION
132274          OD
132275  OUT:    DILGFLAG BZERO DILSTART BZERO DILBOK=:DILGFLAG
132301          "PMFIX"; *MON 2UNFI
132303          *MON 2RTEX
132304  RBUS
132316
132316  *'BBACS
132316   %=============================================================================
132316  %     B A C K R O U N D   P R O G R A M  A L L O C A T I O N   S Y S T E M
132316  %
132316  SUBR PT3MBAPROC,9BPTMOUT
132316
132316  %=============================================================================
132316  %               C F B P P R O C
132316  %
132316  % LOCAL SUBROUTINE TO SEARCH FOR A BACKGROUND PROCSESS TO ALLOCATE
132316  %
132316  % ENTRY:        B=INPUT DATAFIELD
132316  %
132316  % EXIT:         NO BACKGROUND PROCESS FOUND
132316  %
132316  % EXIT+1:       BACKGROUND PROCESS FOUND.
132316  %               X=ADDRESS IN BACKGROUND PROCESS TABLE (SBPRTAB)
132316  %
132316  INTEGER BFOUND
132317  "EBPPROC": 0=:BFOUND
132320              FOR X:="SBPRTAB" STEP BPRTSIZE TO "EBPRTAB"-BPRTSIZE DO
132325                  IF X.CBPTERM=B THEN EXITA FI          % BACKGROUND PROC FOUND
132331                  IF A=0 AND BFOUND=0 THEN              % BACKG.PROC. IS FREE AND NO OTHER CANDIDATE IS FOUND
132334                      IF X.BPRFLG/\BTYPRMASK><0 THEN    % LEGAL FOR THIS DEVICE TYPE?
132337                          X=:BFOUND                     % FREE ENTRY FOUND
```

```
132340                  FI
132340                FI
132340   NXT:       OD
132342              IF X:=BFOUND><0 THEN EXITA FI          % A FREE BACKGROUND PROCESS IS FOUND
132345              EXIT                                   % NO BACKGROUND PROCESS FOUND
132346   *)FILL
132351
132351   INTEGER POINTER LREG
132352   INTEGER CADDR,PRVTADDR
132354
132354   %=============================================================================
132354   %       P T 3 M B A P R O C
132354   %
132354   % SUBROUTINE TO CONNECT A TERMINAL (TAD) TO A BACKGROUND PROCESS
132354   %
132354   % MONITOR LEVEL
132354   %
132354   % ENTRY:      B=INPUT DATAFIELD TO CONNECT TO A BACKGROUND PROCESS
132354   %
132354   % EXIT:       NO FREE BACKGROUND PROCESS FOUND
132354   %
132354   % EXIT+1:     THE CONNECTION BETWEEN THE TERMINAL (TAD) AND
132354   %             A BACKGROUND PROCESS IS SET UP.
132354   %             X=BACKGROUND PROGRAM (RT-DESC.)
132354   %
132354   DISP 12; DOUBLE 500TU; PSID  % ND-500 CPU TIME USED (IN ND-500 PROC.DESCRIPTION)
132354
132354   PT3MBAPROC:
132354          A:=L=:"LREG"
132356          CALL CFBPPROC; GO NFOUND                  % TRY TO FIND FREE BACKGROUND PROC.
132360   OK:    A:=B=:X.CBPTERM                           % ALLOCATE BACKGROUND PROCESS
132362          X=:CADDR                                  % ADDR OF ELEMENT IN SBPRTAB
132363          X.BPRFLG/\BPRCLMSK=:X.BPRFLG              % CLEAR SESSION DEPENDENT FLAGS
132366          X.BBPROC.STATUS BZERO 5WAIT BZERO 5REP=:X.STATUS
132373          "9ENTOPCOM"=:X.STADR                      % START ADDR. OF BACKGROUND PROGRAM
132375          X=:D:=X.RTDLGADDR                         % X=ADDR OF REGISTER BLOCK
132377          T:=0; A:=B; *AAX DBREG; STATX             % SET REGISTERS FOR BACKGROUND PROGRAM
132403          X:=A; A:=D; T:="DBPROG"; CALL XSTDFADDFR  % UPDATE DBPROG IN DATAFIELD
132407          CALL 9GTLOGDV; D:=0                       % FIND LOGICAL DEVICE NUMBER OF TERMINAL (TAD)
132411          A:=D=:CADDR.BPLOGDV
132414          X:=B; CALL GBTINDX; GO NSKP; A=:CADDR.BTBINDX   % FIND INDEX IN BACKGROUND TABLE
132421          A*5PRVTSIZE+"PRVTTABLE"=:X=:PRVTADDR      % ADDRESS OF PRIVATE-SYSTEM-SEGMENT-INFO TABLE
132425          X.SVBPRFLG BONE BPSOK=:CADDR.BPRFLG       % INSERT FLAG BITS
132431          X:=X.BBPROC; CALL GTSLPINDEX; GO NOTSLICED; X=:D   % GET INDEX OF BACKG.PROG.'S INDEX IN TIMESLICE
132435          PRVTADDR.SVTSLSTAT BONE 5ESCF=:TSLSTATUS(D)        % SET TIMESLICE CLASS
132442   NOTSLICED: A:=0=:D=:CADDR.TTMCOUNT                % CLEAR REST OF ELEMENT IN SBPRTAB
132446          AD=:X.DIOUVAL=:X.1TUSED=:X.5TUSED
132451          0=:X.TTMCOUNT
132452          A:=X.BBPROC-"BAK01"=:D:=0; T:=5RTSIZE; *RDIV ST
132460          X:="XBCKTABLE"; X+A; T:=0; A:=B; *STATX          % UPDATE XBCKTABLE
132465          X:=CADDR.BBPROC
132467          MIN "LREG"
132470   OUT:   GO LREG
132471   NSKP:  X:=B; GO OUT
132473
132473   % NO BACKGROUND PROCESS FOUND, CHECK IF ANY
132473   % OF THE BACKGROUND PROCESSES IS FREE
132473   NFOUND: FOR X:="SBPRTAB" STEP BPRTSIZE TO "EBPRTAB"-BPRTSIZE DO
132500              X=:D
132501              IF X.BPRFLG NBIT BPCFIXED THEN          % NOT PERMANENT OCCUPIED
```

```
132504                  IF X.BBPRO.TLINK=0 AND X.WLINK=0 THEN
132511                     0=:D.CBPTERM                          % BACKGROUND PROCESS IS PASSIVE
132513                  FI
132513               FI; X:=D
132514            OD; CALL FAR CFBPPROC; GO ERR                  % CHECK IF ANY FREE NOW
132520            GO OK
132521
132521   ERR:    CALL 9GTLOGDV; GO NSKP
132523           A:=D; CALL 9ERR(#98); GO NSKP
132527   *)FILL
132547
132547   %-----------------------------------------------------------------------------
132547   %       X S T Y P R I N G   -   X R T Y P R I N G
132547   %
132547   % XSTYPRING: LOCAL SUBROUTINE TO SAVE TYPRING AND THEN SET BIT 5NORESERV IN TYPRING
132547   % XRTYPRING: LOCAL SUBROUTINE TO UNSAVE THE SAVED TYPRING
132547   %
132547   INTEGER LRG,OTYPRING
132551   XSTYPRING:
132551   @LIB CXCPU
132551               IF CBPTERM.TYPRING BIT 5TERM THEN A:=X+"9CXTI" ELSE X.DFOPP FI
132561   @ELIB
132561   @LIB CXCPU-,
132561               X:=A;X.TYPRING=:OTYPRING BONE 5NORESERV=:X.TYPRING
132566               IF A BIT 5TERM THEN
132570                  T:=L=:LRG:="TYPRING"; CALL XSTDFADDR; LRG=:L
132576               FI; EXIT
132577   XRTYPRING:
132577   @LIB CXCPU
132577               IF CBPTERM.TYPRING BIT 5TERM THEN A:=X+"9CXTI" ELSE X.DFOPP FI
132607   @ELIB
132607   @LIB CXCPU-,
132607               X:=A; OTYPRING=:X.TYPRING
132612               IF A BIT 5TERM THEN
132614                  T:=L=:LRG:="TYPRING"; CALL XSTDFADDR; LRG=:L
132622               FI; EXIT
132623
132623   %-----------------------------------------------------------------------------
132623   %             C P R L O G O U T
132623   %
132623   % LOGOUT BACKGROUND PROGRAM
132623   %
132623   INTEGER BRG
132624   CPRLOGOUT: A:=L=:LRG
132626           *IOF
132627           X:=CBPTERM
132630   @LIB CXCPU
132630           IF X.TYPRING BIT 5TERM THEN
132633              T:="FLAGB"; CALL XGTDFADDR; A/\ESCMASK; T:="FLAGB"; CALL XSTDFADDR
132640              T:="LAST"; A:=-1; CALL XSTDFADDR
132643              T:="DFLAG"; CALL XGTDFADDR; A BZERO 5IESC; T:="DFLAG"; CALL XSTDFADDR
132650              X:=:B=:BRG; "STESCAPE"; CALL PRLOGOUT
132654           ELSE
132655              X:=:B=:BRG; FLAGB/\ESCMASK=:FLAGB
132662              -1=:LAST; DFLAG BZERO 5IESC=:DFLAG
132667              "ESCAPE"; CALL PRLOGOUT
132671           FI
132671   @ELIB
132671   @LIB CXCPU-,
132671           BRG=:B; LRG=:P
```

```
132675
132675
132675    %=============================================================================
132675    %             C T X T O U T
132675    %
132675    % LOCAL SUBROUTINE TO OUTPUT A TEXT STRING
132675    %
132675    % ENTRY:      A=ADDRESS OF TEXT STRING TERMINATED BY '
132675    %
132675    CTXTOUT: A=:D; X:=0
132677           DO
132677              T:=D; *LBYT
132701           WHILE A><##'
132704              T:=BPLOGDV; *MON 20UTB; JMP *+1
132707              X+1
132710           OD; EXIT
132712    *)FILL
132725
132725
132725    %=============================================================================
132725    %             9 B P T M O U T
132725    %
132725    % RT PROGRAM TO LOGOUT BACKGROUND PROCESSES
132725    %
132725    % THE PROGRAM IS RUNNING WITH AN INTERVAL OF 15 SECS.
132725    %
132725    INTEGER 1COUNT=?,2COUNT=?,CINDX=?,5TU1=?,5TU2=?,CDTI1=?,CDTI2=?                 --
132725    DOUBLE DCOUNT=?,D5TUSE=?,CDTINT=?
132725
132725    INTEGER POINTER PVALUE:=VALUE,PCBYTP:=CBYTP,PXVALUE:=XVALUE
132730    INTEGER ARRAY POINTER PCONST:=CONST
132731    INTEGER POINTER CLRET
132732    CSUBR: A:=L=:"CLRET"
132734           ONTTMCOUNT-TTMCOUNT=:PVALUE
132737           A=:D:=0; T:=4; *RDIV ST                          % 4 ACTIVATIONS OF BPTMP PER MINUTE
132743           A=:PXVALUE; 0=:PCBYTP
132745           FOR X:=-5 DO                                   % COMPUTE HOW MANY MINUTES LEFT
132746              A:=PXVALUE=:D:=0; T:=PCONST(X); *RDIV ST
132753              T:=D=:PXVALUE
132755              IF A><0 OR X=-1 OR T:=PCBYTP><0 THEN
132764                 A+60; X=:D:=PCBYTP; T:="XTMLEFT"; *SBYT
132771                 X+1=:PCBYTP:=D
132774              FI
132774           OD; X:=PCBYTP=:D:=0
133000           DO
133000              T:="TXMINS"; *LBYT
133002           WHILE A><##'
133005              X+1:=:D; T:="XTMLEFT"; *SBYT
133011              X+1:=:D
133013           OD
133014           IF PVALUE><4 THEN                        %IF =4  THEN ONE MINUTE LEFT, NO 'S'
133020              ##S; X:=D; T:="XTMLEFT"; *SBYT
133024              D+1
133025           FI; X:=D; ##'; T:="XTMLEFT"; *SBYT
133031           CALL FAR XSTYPRING
133032           IF BPRFLG NBIT BPWARN THEN
133035              A BONE BPWARN=:BPRFLG; "TXCRLF"; CALL FAR CTXTOUT
133041           FI; "TXWARN"; CALL FAR CTXTOU                  % GIVE WARNING
133043           "TXXBELL"; CALL FAR CTXTOUT
133045           "TXBLA";   CALL FAR CTXTOUT; CALL FAR XRTYPRING
```

```
133050          GO CLRET
133051    *)FILL
133063
133063    BBPTMOUT: X:="SBPRTAB"
133064    LOOP:   IF X>>="EBPRTAB" GO FAR FINI                          % SEARCH ALL ELEMENTS IN TABLE FOR TIMOUT
133067          X=:CINDX=:B
133071          IF BPRFLG NBIT BPRTMOUT OR CBPTERM=0 GO FAR EDO       % NOT TIMOUT ON THIS ELEMENT
133077          IF A.RTRES=0 OR A.ACTPRI NBIT 5BACKGR GO FAR EDO      % NOT USED OR RESERVED BY RT-PROGRAM
133106          IF X.WLINK=0 AND X.TLINK=0 GO FAR EDO                 % BACKGROUND PROGRAM IS PASSIVE
133113          IF X><BBPROC GO FAR EDO
133116          AD:=X.DTINT=:CDTINT                                   % ND-100 CPU TIME USED
133120          X:=CBPTERM
133121          T:="FYLLE"; CALL XGTDFADDR; A=:1COUNT
133124          T:="HENTE"; CALL XGTDFADDR; A+1COUNT=:1COUNT          % INPUT CHECKSUM
133130          90UVAL=:2COUNT
133132    *)8BACS 8N500
"133132           0=:5TU1=:5TU2; BBPROC=:D; CALL FSEMA; GO NO500       % CHECK IF PROGRAM USING ND-500
133140          X.500TU=:D5TUSED                                      % ND-500 TIME USED
133142    *)8BACS
"133142    NO500: AD:=1TUSED; IF D><CDTI2 OR A><CDTI1 GO FAR SAMPLE   % ANY ND-100 CPU TIME USED?
133151          AD:=DIOUVAL; IF D><2COUNT OR A><1COUNT GO FAR SAMPLE % ANY TERMINAL I/O?
133160    *)8BACS 8N500
"133160           AD:=5TUSED; IF D><5TU2 OR A><5TU1 GO FAR SAMPLE     % ANY ND-500 CPU TIME USED?
133167    *)8BACS
"133167           GO MBTMOUT; *)FILL
133201
133201    INTEGER 1COUNT,2COUNT,CINDX,5TU1,5TU2,CDTI1,CDTI2
133210    DOUBLE DCOUNT=1COUNT,D5TUSE=5TU1,CDTINT=CDTI1
133210    INTEGER VALUE,XVALUE,CBYTP
133213    @DEC
133213    DATA(10000,1000,100,10,1); INTEGER ARRAY CONST(0)
133220    @OCT
133220
133220    INTEGER CTLREP
133221    MBTMOUT: MIN TTMCOUNT                                         % NO ACTIVITY, INCREMENT PASSIVE TIME
133222          X:=CBPTERM; T:="BSTATE"; CALL XGTDFADDR
133225          IF A=5LOGIN THEN                                      % NOT LOGGED IN
133230             CALL FAR XSTYPRING; A:="TXBELL"; CALL FAR CTXTOUT % GIVE BELL
133233             CALL FAR XRTYPRING
133234             IF TTMCOUNT>>=NONTTMCOUNT THEN CALL FAR CPRLOGOUT FI  % LOGOUT
133241          ELSE
133242             IF TTMCOUNT>>=TTMWARNING THEN
133246                X:=CBPTERM; T:="FLAGB"; CALL XGTDFADDR; A/\1=:CTLREP
133253                IF A=0 AND ONTTMCOUNT-TTMCOUNT<<4 THEN          % GIVE BELL EATCH TIME IN LAST MINUTE BEFORE LO
133261                   CALL FAR XSTYPRING; "TXBELL"; CALL FAR CTXTOUT
133264                   CALL FAR XRTYPRING
133265                FI
133265                IF TTMCOUNT>>=ONTTMCOUNT THEN
133271                   IF CTLREP><0 THEN
133273                      X:=CBPTERM; T:="FLAGB"; CALL XGTDFADDR      % THE RUNNING PROGRAM HAS THE REPONSIBILITY
133276                      A BONE 5LOGOUT; T:="FLAGB"; CALL XSTDFADDR  % TERMINATE
133301                   ELSE
133302                      CALL FAR CPRLOGOUT                        % LOGOUT USER
133303                   FI; GO EDO
133304                FI
133304                IF TTMCOUNT/\3=0 THEN CALL FAR CSUBR FI         % GIVE WARNING EACH MINUTE
133310             FI
133310          FI
133310    EDO:  X:=CINDX+BPRTSIZE; GO FAR LOOP
133313
```

```
133313   FINI:  *MON 2RTXT
133314
133314   % PROGRAM IS ACTIVE, SAMPLE NEW VALUES TO TEST AGAINST NEXT TIME
133314   SAMPLE: BBPROC.DTINT=:1TUSED; DCOUNT=:DIOUVAL
133321           0=:TTMCOUNT
133322   *"8BACS 8N500
"133322           D5TUSED=:5TUSED
133324   *"8BACS
"133324           BPRFLG BZERO BPWARN=:BPRFLG
133327           GO EDO
133330   *)FILL
133350
133350   INTEGER TXCRLF(0); *6412; 6412; #''
133353   INTEGER TXBELL(0); *3407; #''
133355   INTEGER TXWARN(0); *3407; 3407; 3415
133360   *'IF NO ACTIVITY ON THE TERMINAL, YOU WILL BE LOGGED OUT IN  '
133416   * *-1/
133415   INTEGER XTMLEFT(0); *'                '
133425   INTEGER TXMINS:=' MINUTE'
133431   INTEGER TXXBEL(0); *3407; 3407; #''
133434   INTEGER TXBLA:='  '
133436   RBUS
133436
133436   @MAC

)9SCLC
133436  %=============================================================================
133436  %             S B P R T A B
133436
133436  )MCDEF 9BPEX $BPNN
       $BPNN ;0;0;0;61;0;0;0
       0;0;0;0;0;0
]
133436
133436  "
133436  SBPRT=*
133436  "8BACS
133436          BAK01;0;0;DT01R;62;0;0;0
133446          0;0;0;0;0;0
133454  "8BP5 8BACS; 9BPEX BAK05
133454          BAK05;0;0;0;61;0;0;0
133464          0;0;0;0;0;0
133472
133472  "8BP6 8BACS; 9BPEX BAK06
133472          BAK06;0;0;0;61;0;0;0
133502          0;0;0;0;0;0
133510
133510  '
133510  EBPRT=*
133510
133510  "8BACS
133510  )9RCLC
)9SLPL133510    *"
"133510
133510    *"8PTS
"133510   %=============================================================================
133510   %       P R V T T A B L E
133510   %
133510   % TABLE FOR SAVING VARIABLES FROM THE SYSTEM SEGMENT
133510   % USED BY BACKGROUND PROCESS ALLOCATION SYSTEM
```

```
133510    %
133510    INTEGER PRVTTABLE(0)
133510
133510    @MAC

)9SCLC
133510   )MCDEF MPRVT $FLG
0;0;0;12;0;0;$FLG ;0
]
133510
133510   "8BACS
133510          0;0;0;12;0;0;62;0
133520   "8TR5 8BACS; MPRVT 61
133520   0;0;0;12;0;0;61;0
133530
133530   "8TR6 8BACS; MPRVT 61
133530   0;0;0;12;0;0;61;0
133540
133540   "BAD01 8BACS; MPRVT 60
133540   0;0;0;12;0;0;60;0
133550
133550   "
133550
133550   )9RCLC
)9SLPL133550    *"
"133550    INTEGER EPRVTTABLE(0)
133550
133550    @EOF
133550
```

```
133550   %
133550   %=============================================================================
133550   %  PIT3-PITO-CODE
133550   %
133550   %=============================================================================
133550   *"-3PIT
"133550   %
133550   *"8UDMA 8PUDM
"133550   *)KILL 3PITU 3POFU
133550   *"8UDMA+8VICO -3PIT
"133550
133550   *"8PUDM+8PVIC -3PIT
"133550   *"3PITU+3POFU
'133550
133550
133550   *"8PIOC 8PPIO
"133550   *)KILL 3PITP 3POFP
133550   *"8PIOC 3PIT
"133550   * 3PITP;*-1/        % PIOC ON PIT3 SEGMENT
133550   *"
"133550   *"8PPIO -3PIT
"133550   *"8PIOC+8PPIO -3PIT
"133550   *"3PITP+3POFP
"133550   %=============================================================================
133550   %           M O N P I    E X E L    P I T I M    P I O R E
133550   %
133550   %       MONITOR CALL PIOC
133550   %
133550   %=============================================================================
133550   %
133550   %
133550   %       THE MONITOR CALL  P I O C  ON PITO OR PIT3
133550   %
133550   %       PURPOSE:                USED FOR COMMUNICATION WITH PIOC
133550   %
133550   %       ARGUMENTS:              X - DEMANDFIELD, B - PARAMPOINTER
133550   %                               PT 0, APT = USERS APT, ION, PON
133550   %
133550   %       CALLING SEQUENCE:       T - FUNCTION
133550   %                               X - PIOC LDN
133550   %                               A - PARAM POINTER
133650   %                               IF T = -1 THEN
133550   %                                  A ------> INSTRUCTION TO BE EXECUTED
133550   %                                            KEY FOR SECURITY
133650   %                                            X-REG
133550   %                                            T-REG
133550   %                                            A-REG
133550   %                                            D-REG
133550   %                                 ELSE
133550   %                                  PARAMETERS IN REG (SEE INDIVIDUEL COMMANDS)
133550   %                               ENDIF
133550   %
133550   %       RETURN:                 STATUS IN T
133550   INTEGER DEMSV                                          % PONTER TO DEMFIELD
133551   SUBR APIOCM,PIRET,PIOCE,EXEL
133551   APIOCM:
133551          X=:DEMSV
133552
133552          IF X.ZTREG = -1 THEN GO EXEL FI                 % SPECIAL FUNCTION (EXECUTE L-REG) NO RETURN
133557          CALL GETO
```

```
133560          *"3PITP
"133560          CALL S3NOALPIT                                  % PARAMETERS TO MONITOR CALL ARE ON PIT 3
133561          *"3POFP+3PITP
"133561          MLEV; *MST PIE                                  % IN CASE OF TIMEOUT
133563          X.ZXREG; CALL LOGPH                             % FIND DATAFIELD
133565          IF A = 0 THEN D=:A FI                           % ACCEPT BOTH INPUT AND OUTPUT
133567          IF A = 0 THEN PE1; GO PIRET; FI                 % ILLEGAL LDN
133572          A=:B
133573          IF PINIT >< 173 THEN PE1; GO PIRET; FI          % THIS IS NOT A PIOC DATAFIELD
133601          IF DEMSV.ZTREG >> PMFUN THEN PILF; GO PIRET; FI % ILLEGAL FUNCTION
133610          *"3POFP;
"133610          GO PIOCE                                        % GO TO POF OR PIT3 AREA
133611
133611  %       RETURN FROM MONITOR CALL
133611  %       RETURN STATUS IN A-REG
133611
133611  PIRET: *BSET ZRO
133612          X := DEMSV =: B:=RTRES
133615          A=: ZTREG; *IOF
133617          CALL BRELEASE; CALL RET
133621  RBUS
133634
133634
133634  %       SPECIAL FUNCTION   EXECUTE INSTRUCTION AND GIVE TADX BACK TO USER
133634
133634
133634  SUBR EXEL
133634  %=============================================================================
133634  % SPECIAL MON PIOC FUNCTION
133634  %=============================================================================
133634  INTEGER POINTER PPAST:=PASSTYPE
133635  INTEGER PARP,HPASS
133637  TRIPLE TADR
133642  EXEL:
133642            A:=B=:PARP                          % SAVE PARAM POINTER
133644            CALL ALTOFF                         % ACCSESS SYSTEM SEG ON PIT0
133645            A:=PPAST;*BSET ZRO
133647            A=:HPASS;  B=:D;X=:B
133652            CALL ALTON;D=:B                     % ACCESS USER PARAMETERS
133654            CALL GET6; * BSET ZRO
133656            IF BACKG = 0 OR X.OLDPAG/\3-1>0 OR HPASS = 2 THEN % RT OR RING 2 OR SYSTEM
133671                  IF X.D0 /\ 177700 = 143300 THEN                % SEGMENT MUST BE FIXED BEFORE LOAD A
133676                     X.ZXREG; CALL LOGPH                          % FIND DATAFIELD
133700                     IF A = 0 THEN D=:A FI                        % ACCEPT BOTH INPUT AND OUTPUT
133702                     IF A = 0 THEN PE1; GO PIRET; FI              % ILLEGAL LDN
133705                     A=:B
133706                     IF X.D1 >< PIKEY THEN PNOTP; GO PIRET FI      % CHECK KEY IN USER PARAM
133714                     *"3POFP
"133714                     CALL FIXOK
133715                     *"3POFP
"133715                   FI
133715                  X:=DEMSV=:B
133717                  X.D0=:L; * LDF D3,X
133722                  X:=X.D2; * EXR SL
133724                  TAD =:TADR
133725                  CALL ALTON; MLEV; *MST PIE
133730                  * BSET ZRO 0
133731                  A:=PARP=:B:=DEMSV=:L; TAD:=TADR
133736                  * BSET ONE 0
133737                  A=:P4; T=:P3; A:=D=:P5; X=:P2
```

```
133744                  MLEV; *MCL PIE
133746                  L=:B; CALL ALTOFF
133750                  A;=POK
133751              ELSE
133752                 A:=PNOTP
133753              FI
133753              GO FAR PIRET
133754    RBUS
133770
133770    %       SEGNR IN A, PAGE IN D
133770    SUBR PM116,PM160
133770    INTEGER SEG,PAGE,I
133773    DOUBLE SEGPA=SEG
133773    INTEGER PARS;=(SEG,PAGE)
133775    PM116: A=:SEG; "PARS";
133777           *"3POFP
"133777           * MON 116;
134000           0=:I
134001           FOR I STEP 1 TO 20 DO
134005             A:=PXT(I)
134007             IF A >< 0 THEN                    % ANY XT BLOCK ?
134010               A=:D; 0=:PXT(I)                 % YES, DISCONNECT AT LEVEL 12
134013               "SLV12"; *IOF; IRW LV12B DP
134016               A:="DIS12"; *IRW LV12B DT
134020               A:=D; *IRW LV12B DL
134022               LV12; *MST PID; PION
134025             FI
134025           OD
134031           *"3POFP
"134031           EXIT
134032    PM160: AD=:SEGPA; "PARS";
134034           *"3POFP
"134034           * MON 160;
134035           *"3POFP
"134035           EXIT
134036    RBUS
134042    *"
"134042
134042    *"8PIOC+8PPIO -3PIT
"134042    *"8PIOC+8PPIO -3PIT
"134042
134042    *"8PIOC+8PPIO -3PIT
"134042    *"-3PIT
"134042    @EOF
134042
```

```
134042   %
134042   %=============================================================================
134042   %  PIT3-POF-CODE
134042   %
134042   %=============================================================================
134042   %
134042   *!KILL 3PITU 3POFU
134042   *"8UDMA+8VICO 3PIT
134042    *  3PITU;*-1/       % UDMA ON PIT3 SEGMENT
134042   *"
134042    *"8PUDM+8PVIC -3PIT
134042
134042   *"3PITU+3POFU
134042    "MAC

)9SCLC
134042  %
134042  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134042  %%  DRIVER FOR ND852 UNIVERSAL DMA INTERFACE
134042  %%
134042  %%
134042  %%  DRIVER TO BE CALLED WITH:
134042  %%
134042  %%  T = FUNCTION CODE,   PARAMETER (D AND X  REG)        INN     /  OUT
134042  %%        0  : DMA INPUT I.E. READ FROM INTERFACE  *    LENGTH      ACTUAL LENGTH
134042  %%        1  : DMA OUTPUT I.E. WRITE TO INTERFACE  *    LENGTH           -
134042  %%        2  : SAME AS 0 BUT IMMEDIATE RETURN      *    LENGTH
134042  %%        3  : SAME AS 1 BUT IMMEDIATE RETURN      *    LENGTH           -
134042  %%        7  : TEST MODE                                  -          HARDW. STATUS
134042  %%        20 : ND852 STATUS READ                          -          HARDW. STATUS
134042  %%        21 : DEVICE CLEAR                               -               -
134042  %%        24 : LAST STATUS FROM DATAFIELD                 -          HARDW. STATUS
134042  %%             (HANDLED BY TRANSFER ROUTINE)
134042  %%        54 PIO INPUT  WITHOUT INTERUPT          **    USCO PIOD
134042  %%        55 PIO OUTPUT WITHOUT INTERUPT          **    USCO PIOD
134042  %%        56 : PIO INPUT                          **    USCO PIOD
134042  %%        57 ; PIO OUTPUT                         **    USCO PIOD
134042  %%        62 ; WAIT ON INTERUPT/DMA FINISH              1/0          HARDW. STATUS
134042  %%        64 ; ENABLE ATTENTION INTERRUPT
134042  %%        65 ; DISABLE ATTENTION INTERRUPT
134042  %%        70 : USER CONTROL LINES ARE WRITTEN INTO THE  Us.st.lin
134042  %%             CONTROL REGISTER BITS 8 - 15 WITH
134042  %%             BITS 0 - 7 ALL ZERO
134042  %%
134042  %%
134042  %%       MEMORY ADDRESS IN MEMAD AND MEMAD + 1
134042  %%     *  LENGTH IS DOUBLE WORD IN D AND X-REG
134042  %%    **  PARAMETER IN D AND X-REG   DATA FOR PIO OUTPUT IN X-REG.
134042  %%       DATA FROM PIO INPUT IN INDAT (AERRB).
134042  %%       USER CONTROLL LINES TO BE WRITTEN TO CONTROLL WORD IN D-REG.
134042  %%
134042  %%       IN STATUS REGISTER (HSTAT)
134042  %%       I-----------------------------------I
134042  %%       I ....  9  8  7  6  5  4  3  2  1  0 I
134042  %%       I----------I------------------------I
134042  %%       I SET BY    I BIT 0-7 FROM HARDWARE STATUS REG.
134042  %%       I SOFTW.    I BIT 4 : ERROR/ABORT INDICATOR SET BY HARDW. AND SOFTW.
134042  %%
134042  %%       BIT 8 IS THE ATTENTION  INTERRUPT INDICATOR
134042  %%       BIT 9 IS THE TIMEOUT INDICATOR
```

```
134042  %%
134042  %%        IN RETURN PARAMETER WHEN FUNCK >< 0
134042  %%        I-------------------------------------------------------------I
134042  %%        I .... 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0 I
134042  %%        I-----------I-------------------------------------------------I
134042  %%        I LEFT BYTE I    BIT 0-15 FROM HARDWARE STATUS REGISTER
134042  %%        I FROM HSTATI
134042  %%
134042  %%        BIT 16 IS THE ATTENTION  INTERRUPT INDICATOR
134042  %%        BIT 17 IS THE TIMEOUT INDICATOR
134042  %%
134042  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134042  %%        DEFINITIONS FOR IOX INSTRUCTIONS
134042  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134042  RMAR=0        % READ PRESENT MEMORY ADDRESS REGISTER
134042  LMAR=1        % LOAD MEMORY ADDRESS REGISTER
134042  RDAT=2        % READ DATA INPUT REGISTER
134042  LDAT=3        % LOAD DATA OUTPUT REGISTER
134042  RSTAT=4       % READ STATUS REGISTERS
134042  LCONT=5       % LOAD CONTROL WRITE REGISTER
134042  LWCNT=7       % LOAD WORD COUNTER REGISTER
134042  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134042  %%
134042  %%        MAIN DRIVER
134042  %%
134042  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134042  % MACRO TO EXECUTE IOX-INSTR DATA IN A-REG
134042  )MCDEF UIOX  $FUNIX

        LDT   HDEV ,B
        AAT  $FUNIX

        IOXT

|
134042  %
134042  %   MAIN DRIVER
134042  ND852,  STF I (9TREG          % SAVE T A & D REG
134043          STX I (9XREG          % SAVE X
134044          COPY  SL DA
134045          STA I (9LREG
134046          UIOX  RSTAT
134046
134046          LDT   HDEV ,B
134047          AAT   RSTAT
134047
134050          IOXT
134051
134051
134051          STA   USTAT ,B        % SAVE IN DATA FIELD
134052          JAP   *+2             % ATTENTION INT NOT OCCURED
134053          STA   ATINT ,B        % SET ATTENTION INT OCCURED
134054          AND   (377            % MASK OFF USER STATUS BITS
134055          STA   HSTAT ,B        % SAVE IN DATA FIELD
134056  SWTST,  LDA   SOFTA,B         % SOFTWARE ACTIVATED ?
134057          JAF   FUNCTEST        % YES TEST FUNCTION
134060          LDA   BUSFL ,B        % DMA OR PIO INTERRUPT ?
134061          JAF   *+2             % JMP IF YES
134062          JMP I (EBORT          % NO TEST FOR ABORT
134063          AAA   -1
```

```
134064            JAF    NOTDMA
134065            LDA    HSTAT ,B        % PICK UP STATUS
134066            BSKP   ONE 20 DA       % DMA INTERRUPT ?
134067            JMP I  (DMACT          % JMP IF YES
134070  NOTDMA,   AAA    -1              % PIO INPUT INTERRUPT ?
134071            JAF    *+2
134072            JMP I  (PIOININT       % JMP IF YES
134073            AAA    -1              % PIO OUTPUT INTERRUPT ?
134074            JAF    *+2
134075            JMP I  (PIOOUTINT      % JMP IF YES
134076            AAA    -1
134077            JAF    *+2
134100            JMP I  (WODINT         % WAIT ON INTERUPT INTERUPT
134101            JMP I  (BUSYEX         % BUSY EXIT
134102  %
134102  %         DMA NOT ACTIVE, TEST FUNCTION FROM USER
134102  %
134102  FUNCTEST, STZ    SOFTA ,B       % CLEAR SOFTA
134103            LDA I  (9TREG          % FUNCTION
134104            BSKP   ZRO 170 DA
134105            JMP I  (ERRFU          % ILLEGAL FUNC
134106            JAF    *+2
134107            JMP I  (DMAI           % FUNC = 0 DMA INPUT
134110            AAA    -1
134111            JAF    *+2
134112            JMP I  (DMAUT          % FUNC = 1 DMA OUTPUT
134113            AAA    -1
134114            JAF    *+2
134115            JMP I  (XDMAI          % FUNC = 3 DMA INPUT AND NO WAIT
134116            AAA    -1
134117            JAF    *+2
134120            JMP I  (XDMAU          % FUNC = 4 DMA OUTPUT AND NO WAIT
134121            AAA    -4
134122            JAF    *+2
134123            JMP I  (UTEST          % FUNC = 7 TEST MODE
134124            AAA    -11
134125            JAF    *+2
134126            JMP I  (FINI           % FUNC = 20 INTERFACE STATUS READ
134127            AAA    -1
134130            JAF    *+2
134131            JMP I  (DCLEA          % FUNC = 21 CLEAR DEVICE
134132            AAA    -33
134133            JAF    *+2
134134            JMP I  (PIONI          % FUNC = 54 PIO INPUT  WITHOUT INTERUPT
134135            AAA    -1
134136            JAF    *+2
134137            JMP I  (PIONO          % FUNC = 55 PIO OUTPUT WITHOUT INTERUPT
134140            AAA    -1
134141            JAF    *+2
134142            JMP I  (PIOIN          % FUNC = 56 PIO INPUT
134143            AAA    -1
134144            JAF    *+2
134145            JMP I  (PIOUT          % FUNC = 57 PIO OUTPUT
134146            AAA    -3
134147            JAF * +2
134150            JMP I  (WODMA          % FUNC = 62 WAIT ON INTERUPT
134151            AAA    -2
134152            JAF *+2
134153            JMP I  (ENABA          % FUNC = 64 SET ENABLE ATTENTION INTERRUPT
134154            AAA    -1
```

```
134155          JAF *+2
134156          JMP I (DISAA
134157          AAA   -3
134160          JAF *+2
134161          JMP I (USERC
134162          JMP I (ERRFU            % WRONG FUNCTION
134163  )FILL
134215
134215
134215  %-------------------------------------------------------------
134215  %       FUNC 0 & 2 DMA INPUT
134215  %
134215  XDMAI,  SAA   -1
134216          STA   NOWFL ,B          % SET NO WAIT
134217          JMP *+2
134220  DMAI,   STZ   NOWFL ,B
134221          SAT   7                 % DMA INPUT CONTROL
134222          JMP   DMACO
134223  %
134223  %       FUNC 1 & 3 DMA OUTPUT
134223  %
134223  XDMAU,  SAA   -1
134224          STA   NOWFL ,B          % SET NO WAIT
134225          JMP *+2
134226  DMAUT,  STZ   NOWFL ,B
134227          LDT   (207              % DMA OUTPUT CONTR
134230  DMACO,  STZ   NOWL  ,B
134231          STZ   NOWH  ,B          % ZERO NO OF WORD TRANSF
134232          COPY  ST DD             % SAVE CONTROL IN DREG
134233          LDA   HSTAT ,B          % PICK UP STATUS  %------- NEW TEST
134234          BSKP ONE 20 DA          % DMA ACTIVE ?
134235          JMP   DMAOK             % JUMP IF NO
134236          BSET ONE 40 DA          % SET ERROR INDICATOR
134237          STA   HSTAT ,B
134240          JMP I (FINI             % ACTIVATE USER
134241  DMAOK,  LDA   MEMA1 ,B          % UPPER 16 BITS OF MEMORY ADDRESS
134242          UIOX  LMAR              % WRITE CORE ADDR REG UPPER PART
134242
134242          LDT   HDEV ,B
134243          AAT   LMAR
134243
134244          IOXT
134245
134245
134245          LDA   MEMA2,B           % LOWER 16 BITS OF MEMORY ADDRESS
134246          UIOX  LMAR
134246
134246          LDT   HDEV ,B
134247          AAT   LMAR
134247
134250          IOXT
134251
134251
134251          LDA   9DREG
134252          UIOX  LWCNT             % WRITE WORDCOUNTER UPPER BITS
134252
134252          LDT   HDEV ,B
134253          AAT   LWCNT
134253
134254          IOXT
```

```
134255
134255
134255          LDA   9XREG
134256          UIOX  LWCNT
134256
134256          LDT   HDEV ,B
134257          AAT   LWCNT
134257
134260          IOXT
134261
134261
134261          COPY  SD DA           % SET I/O CONTR AND INTERRUPT ENABLE
134262          ORA   ATTNI ,B        % ENABLE ATTENTION INTERRUPT ?
134263          ORA   UCLIN ,B
134264          UIOX  LCONT           % START DMA TRANSFER
134264
134264          LDT   HDEV ,B
134265          AAT   LCONT
134265
134266          IOXT
134267
134267
134267          SAA   1               % SET DMA ACTIVE
134270          STA   BUSFL ,B
134271          LDA   NOWFL ,B        % NO WAIT MODE ?
134272          JAP   *+2
134273          JMP I (FINWA           % YES FINISH
134274          JMP I (BUSYEX
134275  %-------------------------------------------------------------
134275  %       DMA ACTIVE  DRIVER STARTED BY AN EXTERNAL INTERRUPT.
134275  DMACT,  JPL I (NOWTR          % CALCULATE NO OF WORD TRANSFERED
134276          LDA   NOWFL ,B        % NO WAIT MODE ?
134277          JAZ   *+2
134300          JMP I (BUSYEX         % YES BUSYEX
134301          JMP I (FINI
134302  %-------------------------------------------------------------
134302  %  FUNC = 7    TEST MODE
134302  UTEST,  SAA   10
134303          ORA   UCLIN ,B
134304          UIOX  LCONT
134304
134304          LDT   HDEV ,B
134305          AAT   LCONT
134305
134306          IOXT
134307
134307
134307          JMP I (FINI
134310  %-------------------------------------------------------------
134310  %       FUNC 21 CLEAR  852
134310  DCLEA,  JPL   DCLW
134311          STZ   UCLIN,B
134312          STZ   ATTNI,B
134313          STZ   NOWFL ,B        % CLEAR NO WAIT FLAG
134314          JMP I (FINI           % EXIT
134315
134315  % MASK OF ILLEGAL USER CONTROLL LINES IN 9DREG
134315  UCMA,   LDA   9DREG           % DREG= USER CONTROLL LINES FOR PIO INN/OUTPUT
134316          AND   (177400         % MASK OF ILLEGAL BIT
134317          STA   9DREG
```

```
134320          EXIT
134321  % CLERA DEVICE AND WRITE ZERO IN THE WORD COUNTER
134321  DCLW,    SAA   20
134322           UIOX  LCONT
134322
134322          LDT   HDEV ,B
134323          AAT   LCONT
134323
134324          IOXT
134325
134325
134325           SAA   0               % ZERO WORD COUNTERR
134326           UIOX  LWCNT
134326
134326          LDT   HDEV ,B
134327          AAT   LWCNT
134327
134330          IOXT
134331
134331
134331           IOXT
134332           EXIT
134333
134333  %-------------------------------------------------------------
134333  %       FUNC 54 PIO INPUT WITHOUT INTERUPT
134333  PIONI,  JPL   DCLW            % CLERA DEVICE AND WRITE ZERO IN THE WORD COUNTER
134334          JPL   UCMA            % MASK OF ILLEGAL USER CONTROLL LINES IN 9DREG
134335          SAA   104             % BIT 2 = ACRIVATE TRANSFER, BIT 6 = PIO MODE
134336          ORA   UCLIN ,B        % PERMANENT USER CONTROLL LINES
134337          ORA   ATTNI ,B        % ATTENTION INT ENABLE ?
134340          ORA   9DREG           % USER CONTROLL LINE(S) FOR HANDSHAKE ETC.
134341          UIOX  LCONT
134341
134341          LDT   HDEV ,B
134342          AAT   LCONT
134342
134343          IOXT
134344
134344
134344          UIOX  RDAT
134344
134344          LDT   HDEV ,B
134345          AAT   RDAT
134345
134346          IOXT
134347
134347
134347          STA   INDAT ,B        % SAVE PIO DATA
134350          JMP   FINWA           % GET STATUS AND RETURN TO USER
134351  %-------------------------------------------------------------
134351  %       FUNC 55 PIO OUTPUT WITHOUT INTERUPT
134351  PIONO,  JPL   DCLW            % CLERA DEVICE AND WRITE ZERO IN THE WORD COUNTER
134352          JPL   UCMA            % MASK OF ILLEGAL USER CONTROLL LINES IN 9DREG
134353          LDA   (300            % BIT 6 = PIO MODE, BIT 7 = OUTPUT
134354          ORA   UCLIN ,B        % PERMANENT USER CONTROLL LINES
134355          ORA   ATTNI ,B        % ATTENTION INT ENABLE ?
134356          UIOX  LCONT           %
134356
134356          LDT   HDEV ,B
134357          AAT   LCONT
```

```
134357
134360          IOXT
134361
134361
134361          LDA    9XREG           % GET PIO DATA
134362          UIOX   LDAT
134362
134362          LDT    HDEV ,B
134363          AAT    LDAT
134363
134364          IOXT
134365
134365
134365          LDA    (304            % BIT 2 = ACTIVATE TRANSFER
134366          ORA    UCLIN ,B        % PERMANENT USER CONTROLL LINES
134367          ORA    ATTNI ,B        % ATTENTION INT ENABLE ?
134370          ORA    9DREG           % USER CONTROLL LINE(S) FOR HANDSHAKE ETC.
134371          UIOX   LCONT
134371
134371          LDT    HDEV ,B
134372          AAT    LCONT
134372
134373          IOXT
134374
134374
134374          JMP    FINWA           % GET STATUS AND RETURN TO USER
134375
134375  %-------------------------------------------------------------
134375  %       FUNC 56 PIO INPUT
134375  PIOIN,  JPL    DCLW            % CLERA DEVICE AND WRITE ZERO IN THE WORD COUNTER
134376          JPL    UCMA            % MASK OF ILLEGAL USER CONTROLL LINES IN 9DREG
134377          SAA    100             % BIT 6 = PIO MODE
134400          ORA    ATTNI ,B        % ATTENTION INT ENABLE ?
134401          ORA    UCLIN ,B
134402          UIOX   LCONT           % SET PIO MODE
134402
134402          LDT    HDEV ,B
134403          AAT    LCONT
134403
134404          IOXT
134405
134405
134405          SAA    105             % BIT 0 = ENA. INT. ON RFT, BIT 2 = ACTIVATE TRANSF.
134406          ORA    ATTNI ,B        % ATTENTION INT ENABLE ?
134407          ORA    UCLIN ,B
134410          ORA    9DREG           % USER CONTROLL LINE(S) FOR HANDSHAKE ETC.
134411          UIOX   LCONT
134411
134411          LDT    HDEV ,B
134412          AAT    LCONT
134412
134413          IOXT
134414
134414
134414          SAA    2
134415          STA    BUSFL ,B        % SET PIO INPUT ACTIVE
134416          JMP    BUSYEX          % WAIT FOR INTERRUPT
134417  )FILL
134427  %-------------------------------------------------------------
134427  %    PIO INPUT INTERRUPT
```

```
134427  PIOININT,UIOX RDAT          % READ INTERFACE DATA REGISTER
134427
134427          LDT   HDEV ,B
134430          AAT  RDAT
134430
134431          IOXT
134432
134432
134432          STA   INDAT ,B
134433          JMP   FINI
134434  9LREG,  0
134435  9TREG,  0
134436  9AREG,  0
134437  9DREG,  0
134440  9XREG,  0
134441  %
134441  %-----------------------------------------------------------------
134441  %       FUNCTION 57 PIO OUTPUT
134441  PIOUT,  JPL   DCLW            % CLERA DEVICE AND WRITE ZERO IN THE WORD COUNTER
134442          JPL   UCMA            % MASK OF ILLEGAL USER CONTROLL LINES IN 9DREG
134443          LDA   (300            % BIT 6 = PIO MODE, BIT 7 = OUTPUT
134444          ORA   ATTNI ,B        % ENABLE ATTENTION INTERRUPT ?
134445          ORA   UCLIN ,B
134446          UIOX  LCONT           % INIT PIO OUTPUT MODE
134446
134446          LDT   HDEV ,B
134447          AAT   LCONT
134447
134450          IOXT
134451
134451
134451          LDA   9XREG           % PICK UP PIO OUTPUT DATA
134452          UIOX  LDAT
134452
134452          LDT   HDEV ,B
134453          AAT   LDAT
134453
134454          IOXT
134455
134455
134455          LDA   (305            % BIT 0 = ENA. INT. ON RFT, BIT 2 = ACTIVATE TRANSF.
134456          ORA   ATTNI ,B        % ENABLE ATTENTION INTERRUPT ?
134457          ORA   UCLIN ,B
134460          ORA   9DREG           % USER CONTROLL LINE(S) FOR HANDSHAKE ETC.
134461          UIOX  LCONT           % ENABLE PIO OUT
134461
134461          LDT   HDEV ,B
134462          AAT   LCONT
134462
134463          IOXT
134464
134464
134464          SAA   3
134465          STA   BUSFL ,B        % SET PIO OUTPUT ACTIVE
134466          JMP   BUSYEX          % WAIT FOR INTERRUPT
134467  %-----------------------------------------------------------------
134467  %   PIO OUTPUT INTERRUPT
134467  PIOOUTINT,
134470          JMP   FINI
134471  %-----------------------------------------------------------------
```

```
134471  %       FUNC= 62 WAIT ON INTERUPT
134471  WODMA,  LDA   ATINT ,B        % ATTENTION INTERUPT HAS OCCURED
134472          JAF   WFINI
134473          LDA   HSTAT ,B        % PICK UP STATUS
134474          BSKP  ZRO 20 DA       % DMA ACTIVE ?
134475          JMP   WBUSY           % YES JUST EXIT
134476          LDA   9XREG           % WAIT ONLY FOR DMA FINISH ?
134477          JAZ   WFINI           %  YES FINISH
134500          LDA   ATTNI ,B        % IF ATTENTION INTERUPT ENABLET THEN
134501          JAF   WBUSY           %       OK    AND BUSY   EXIT
134502  WFINI,  JMP   FINI            % ELSE  ERROR AND FINISH EXIT; FI
134503  WBUSY,  SAA   4               % SET WAIT ON INTERUPT
134504          STA   BUSFL ,B        %
134505          JMP   BUSYEX          % WAIT ON INTERUPT
134506  %-------------------------------------------------------------
134506  %       WAIT ON INTERUPT INTERUPT
134506  WODINT, LDA   HSTAT ,B
134507          BSKP  ONE 20 DA       %   IF DMA NOT ACTIVE THEN
134510          JMP   FINI            %     ACTIVATE USER
134511          SAA   1               %   ELSE
134512          STA   BUSFL ,B        %     SET DMA ACTIVE IN BUSFL
134513          JMP   FINW            %     ACTIVATE USER WITHOUT DESTROING BUSFL
134514                                %   FI
134514  %-------------------------------------------------------------
134514  %       FUNC = 64 SET ATTENTION INTERRUPT ENABLE
134514  ENABA,  SAA   40              % BIT 5 IN CONTROLL WRITE REGISTER
134515          STA   ATTNI ,B        % SET ATTENTION INTERRUPT ENABLE
134516          UIOX  LCONT           % WRITE CONTROL REGISTER
134516
134516          LDT   HDEV ,B
134517          AAT   LCONT
134517
134520          IOXT
134521
134521
134521          JMP   FINI
134522  %
134522  %-------------------------------------------------------------
134522  %       FUNC = 65 RESET ATTENTION INTERRUPT ENABLE
134522  DISAA,  STZ   ATTNI ,B        % RESET ATTENTION INTERRUPT ENABLE
134523          STZ   ATINT ,B        % RESET ATTENTION INTERRUPT OCCURED
134524          JMP   FINI
134525  %
134525  %-------------------------------------------------------------
134525  %       FUNC = 70 WRITE USER CONTROL LINES
134525  USERC,  LDA   9XREG           % PICK UP CODE FROM USER
134526          AND   (177400         % GET RID OF ILLEGAL BITS
134527          STA   UCLIN ,B        % STORE USER CODE
134530          ORA   ATTNI ,B        % ENABLE ATTENTION INTERRUPT ?
134531          UIOX  LCONT           % WRITE CONTROL REGISTER
134531
134531          LDT   HDEV ,B
134532          AAT   LCONT
134532
134533          IOXT
134534
134534
134534          JMP   FINWA           % READ STATUS AND RETURN TO USER
134535
134535  %-------------------------------------------------------------
```

```
134535  %       BUSY EXIT
134535  BUSYEX, LDA   9LREG
134536          COPY  SA DL
134537          LDF   9TREG           % UNSAVE REGISTERS
134540          LDX   9XREG
134541          EXIT  AD1             % BUSY EXIT
134542
134542  %-------------------------------------------------------------
134542  % ENTRY FROM DMA AND NO WAIT, AND FUNCTIONS WITHOUT INTERUPT
134542  FINWA,  UIOX  RSTAT           % GET AND SAVE STATUS BEFORE RETURNING TO USER
134542
134542          LDT   HDEV ,B
134543          AAT   RSTAT
134543
134544          IOXT
134545
134545
134545          STA   USTAT ,B
134546          AND   (377
134547          STA   HSTAT ,B
134550          JMP   FINW
134551  %-------------------------------------------------------------
134551  %      FINISH  EXIT           % NORMAL ENTRY
134551  FINI,   STZ   BUSFL ,B
134552  FINW,   LDX   HSTAT ,B        % ENTRY FROM  WODINT
134553          LDA   ATINT ,B
134554          JAZ   *+3
134555          BSET  ONE 100 DX      % SET ATTENTION INTERRUPT OCCURED
134556          STX   HSTAT ,B
134557          STZ   ATINT ,B        %CLEAR ATTENTION INT OCCURED FLAG
134560          LDA   9LREG
134561          COPY  SA DL
134562          RINC  DL              % INCREMENT RETURN ADDR TO FINISHED EXIT
134563          LDA   9AREG
134564          EXIT  AD1
134565  )FILL
134571  %-------------------------------------------------------------
134571  %       TEST IF ABORT INTERRUPT OCCURED
134571  EBORT,  LDA   HSTAT ,B
134572          BSKP  ONE 40 DA
134573          JMP   BUSYEX
134574          LDA   NOWFL ,B
134575          JMP   FINI            % YES ACTIVATE USER
134576
134576  %-------------------------------------------------------------
134576  % CALCULATE NO OF WORD TRANSFERED
134576  NOWTR,  UIOX  RMAR
134576
134576          LDT   HDEV ,B
134577          AAT   RMAR
134577
134600          IOXT
134601
134601
134601          COPY  SA DD
134602          UIOX  RMAR
134602
134602          LDT   HDEV ,B
134603          AAT   RMAR
134603
```

```
134604          IOXT
134605
134605
134605          AND    (377            % AD HOLD MEMORY ADDRESS REGISTER
134606          LDT    MEMA1 ,B
134607          LDX    MEMA2 ,B
134610          COPY   CM2 SX DX
134611          COPY   CM1 ADC ST DT
134612          RADD SX DD
134613          RADD   ST DA ADC
134614          STD    NOWH   ,B
134615          EXIT
134616  %-----------------------------------------------------------
134616  %       WRONG FUNCTION CODE FROM USER
134616  ERRFU,  SAA    -1             % WRONG FUNCTION CODE
134617          STA    HSTAT ,B       % SET HW STATUS TO -1
134620          JMP    FINI
134621  %
134621  %       END OF DRIVER
134621  )FILL
134622  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134622  %%
134622  %%        LEVEL 11 ROUTINE TO PERFORM TRANSFER
134622  %%        ON  852 UNIVERSAL DMA INTERFACE INCLUDING
134622  %%        PIO TRANSFERS.
134622  %%
134622  %%  PARAMETER LIST TRANSFERD BY A-REG TO ABSTRANS
134622  %%
134622  %%  INTEGER POINTER FUNCP            % FUNCTION
134622  %%  DOUBLE  POINTER MEMOP            % MEMORY ADDRESS
134622  %%  INTEGER POINTER BLOCP            % NOT USED
134622  %%  DOUBLE  POINTER DATP             % PARAMETER INN/OUT
134622  %%
134622  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134622
134622  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134622  %%        ACTIVATED FROM STDRIV
134622  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134622  UASTR,  SWAP   SX DB
134623  "3PITU
134623          BSET ONE              % PARAMETERS FROM ALT PT.
134624          LDT I,B  P0
134625          LDD I,B  P1
134626          BSET ZRO
134627  "3PITU+3POFU
134627          STD    MEMA1 ,X       % MEMORY ADDRES
134630          STT    CTRG ,X        % FUNCTION CODE
134631  "3PITU
134631          BSET ONE
134632          LDT   ,B  P3
134633          LDD I,B  P3
134634          BSET ZRO              %    24 BIT WORD COUNTER
134635  "3PITU+3POFU
134635          STD    CDRG ,X        %    CDRG= UPPER BITS, CXRG= LOWER BITS
134636                                %    PIO DATA IN CXRG, PIO UCLIN MASK IN CDRG
134636  "3PITU+3POFU
134636          STT    MRETU,X        % ADDRESS OF RETURN PARAMETER
134637          COPY   SX DB          % UNSAVE BASE
134640          LDA    CTRG ,B        % PICK UP FUNCTION
134641          AAA    -24
```

```
134642          JAZ    UFIN             % JUST EXIT STATUS IS ALREADY IN HSTAT ,B
134643          LDA    ,B TTMR          % SET TIMEOUT TIME
134644          STA    ,B TMR
134645  ULOOP,  LDF    CTRG ,B          % PICK UP PARAMETERS
134646          LDX    CXRG ,B
134647          MIN    SOFTA ,B         % SET SOFTWARE ACTIVATED
134650          COPY   SA DA            % JGA SOFTA=0 TRANSFER DONE (DUMMY)
134651          JPL I  TRNSF ,B         % CALL DRIVER
134652          JMP I  ERROR ,B         % ERROR EXIT
134653          JMP I  BUSY ,B          % BUSY EXIT
134654          JMP I  FINIS ,B         % TRANSFER COMPLETE EXIT
134655
134655  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134655  %%       ACTIVATED FROM DRIVER
134655  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134655  UADDR,  SAA    -1               % JGA INSERTED TO TELL ROUTINE
134656          STA    SOFTA ,B         % THAT CALL WAS NOT SOFTWARE INITIATED.
134657          JMP    ULOOP            % CALL TRANSFER AGAIN.
134660  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134660  %%       BUSY / ERROR
134660  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134660  UBUSY,  STF    CTRG ,B          % SAVE PARAMETERS
134661          STX    CXRG ,B
134662          JPL I  (UWT11            % WAIT FOR INTERRUPT.
134663  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134663  %%       FINISH
134663  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% ...
134663  UFIN,   LDA    RTRES ,B         % ACTIVATE USER ?
134664          JAZ    NOONE            % JMP IF NOONE TO ACTIVATE
134665          LDA    CTRG  ,B         % PICK UP FUNCTION
134666          JAF    NOO              % IF FUNC = 0 THEN RETURN NO OF WORD TRANSF.
134667          LDD    NOWH  ,B
134670          JMP    DAPAR
134671  NOO,    AAA    -54              %
134672          JAZ    *+3
134673          AAA    -2
134674          JAF    NOPO             % IF FUNC =54 OR =56 THEN RETURN PIO DATA
134675          LDA    INDAT ,B         %
134676          SAD ZIN SHR 20          % A=:D; A:="0"
134677          JMP    DAPAR            %
134700  NOPO,   LDA    USTAT ,B         %  ELSE RETURN USER STATUS
134701          COPY   SA DD
134702          LDA    HSTAT ,B         % HSTAT HAVE TIMEOUT, ATTINT ETC.
134703          SHA ZIN SHR 10          %
134704  DAPAR=*
134704  "3PITU
134704          BSET ONE
134705          STD    MRETU I,B        % RETURN DOUBLE INTEGER DATA TO USER
134706          BSET ZRO
134707  "3PITU+3POFU
134707          JPL I  (RTACT           % ACTIVATE USER
134710  NOONE,  STZ    ,B TMR           % CLEAR TIMEOUT TIME
134711          JPL I  (UWT11           % GO AND WAIT FOR INTERRUPTS
134712  )FILL
134714  %%
134714  %%       END OF LEVEL 11 TRANSFER ROUTINE
134714  %%
134714  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134714  %%
134714  %%      ROUTINE ON LEVEL 11 TO HANDLE TIMEOUT
```

```
134714  %%
134714  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134714
134714  UDIIT,   UIOX   RSTAT          % READ STATUS
134714
134714           LDT    HDEV ,B
134715           AAT    RSTAT
134715
134716           IOXT
134717
134717
134717           STA    USTAT ,B
134720           JAN    AYES           % ATTENTION INT ?
134721           LDX    ATINT ,B       % PREVIOUS ATTENTION INT ?
134722           JXZ    ANOT
134723  AYES,    BSET   ONE 100 DA     % SET ATTENTION INT OCCURED
134724           JMP    *+2
134725  ANOT,    BSET   ZRO 100 DA
134726           AND    (777
134727           BSET   ONE 40 DA      % SET ERROR INDICATOR
134730           BSET   ONE 110 DA     % SET TIME OUT INDICATOR
134731           STA    HSTAT ,B       % SAVE IN DATAFIELD
134732           STZ    ATINT ,B       % RESET ATTENTION INTERRUPT INDICATOR
134733           LDA    NOWFL,B        % PICK UP STATUS
134734           JAN    UFIN           % NO WAIT MODE, DO NOT STOP DMA
134735           LDA    BUSFL ,B       % PICK UP BUSY INDICATOR
134736           STZ    BUSFL ,B       % CLEAR BUSY INDICATOR
134737           AAA    -4             % IF WAIT ON INTERUPT THEN
134740           JAZ    UFIN           %   DO NOT CLEAR DEVICE; FI
134741           SAA    20             % BIT 4 = DEVICE CLEAR
134742           ORA    ATTNI ,B       % ENABLE ATTENTION INTERRUPT ?
134743           UIOX   LCONT
134743
134743           LDT    HDEV ,B
134744           AAT    LCONT
134744
134745           IOXT
134746
134746
134746           JMP I FINIS ,B        % CONTINUE AS WHEN TERMINATING DRIVER
134747
134747  )FILL
134750  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134750  %% INPUT DATA FIELD LAYOUT
134750  %%  DISP.  SYMBOL  CONTENS
134750  %%  -45 - -40                  % NOT USED
134750  %%  -37     MRETURN            % ADDRESS FOR RETUR PARAMETER
134750  %%  -35     BUSFL              % 0= IDLE, 1=DMA , 2=PIO IN ,3=PIO OUT
134750  %%                             % 4= WAIT ON INTERUPT
134750  %%  -34     NOWFL              % -1 = NO WAIT (FUNC 2 AND 3)
134750  %%  -30     NOWH               % NO OF WORD TRANSFERD UPPER BITS
134750  %%  -27     NOWL               % NO OF WORD TRANSFERD LOWER BITS
134750  %%  -26     CTRG               % FUNCTION CODE
134750  %%  -25     CARG               %
134750  %%  -24     CDRG               % WORDCOUNT UPPER BIT
134750  %%  -23     CXRG               % WORDCOUNT LOWER BIT OR PIO OUTPUT DATA
134750  %%  -22     ATTNI              % =40 ATTENTION INTERUPT IS ENABLED
134750  %%  -21     EXTIO              % NOT USED
134750  %%  -20     USTAT              % UNMASKED STATUS REG.
134750  %%  -17     INDAT              % PIO INPUT DATA
```

```
134750  %%  -16    UCLIN                        % USER CONTROLL LINES
134750  %%  -15    ATTINT                       % >< 0 IF ATTENTION INTERUPT HAS OCCURED
134750  %%                                      % BUT NOT RECIVED BY USER
134750  %%  -14    SOFTA                        %
134750  %%  -12    TRNSF/ND852                  % DRIVER
134750  %%  -11    BUYSY/UBUSY                  % BUSY EXIT FROM DRIVER
134750  %%  -10    FINISH/UFIN                  % FINISHED EXIT FROM DRIVER
134750  %%  - 7    ERROR/UBUSY                  % ERROR EXIT TREATED AS BUSY EXIT
134750  %%  - 6    TMSUB/UDTMO                  % TIMEOUT SUBROUTINE
134750  %%  - 5    TMR                          % TIMEOUT COUNTER
134750  %%  - 4    TTMR/177771                  % CA 6-7 SEC TIMEOUT
134750  %%  - 3    HDEV                         % IOX DEVICE ADDRESS
134750  %%  - 2    STDRIV/USDRV -> PT3US        % START ADDRESS LEVEL 11
134750  %%  - 1    DRIVER/UDDRV -> PT3UD        % DRIVER
134750  %%    2    BWLIN                        % BWLINK
134750  %%    3    TYPRI                        % TYPRING
134750  %%    4    ISTATE                       % NOT USED
134750  %%    6    MFUNC/RETRA                  % RETRANS AS FOR MAG TAPE
134750  %%   11    MTRAN/MTRNS                  % MTRNS AS FOR MAG TAPE
134750  %%   14    MEMA1                        % MEMORY ADDRESS UPPER BIT
134750  %%   15    MEMA2                        % MEMORY ADDRESS LOWER BIT
134750  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
134750  )KILL RMAR LMAR RDAT LDAT RSTAT LCONT LWCNT UIOX SWTST OTDMA CTEST
134750  )KILL XDMAI DMAI XDMAU DMAUT DMACO DMAOK DMACT UTEST DCLEA PIONI
134750  )KILL PIONO PIOIN ININT 9LREG 9TREG 9AREG 9DREG 9XREG PIOUT UTINT
134750  )KILL WODMA WFINI WBUSY ODINT ENABA DISAA USERC USYEX FINI NOAT FINWA
134750  )KILL EBORT NOWLR NOWTR ERRFU ULOOP NOO NOPO DAPAR NOONE
134750  )KILL AYES ANOT PFIN FINW DCLW UCMA
134750  )9RCLC
)9SLPL134750    *"
"134750
134750    *)KILL 3PITP 3POFP
134750    *"8PIOC 3PIT
"134750   *   3PITP;*-1/          % PIOC ON PIT3 SEGMENT ?
134750    *"
"134750   *"8PPIO -3PIT
"134750   *"3PITP+3POFP
```

```
134750   %
134750   %=============================================================================
134750   %              F I X O K    S E G F N    C H P O    P R E A D
134750   %              P W R I T E    K R E A D    P I C A C
134750   %
134750   %              P I O C    N O N R E S I D E N T  (PIT3 OR POF)   P A R T
134750   %              U T I L I T Y    R O U T I N E S
134750   %
134750   %=============================================================================
134750
134750   SUBR PREAD,PWRIT,KREAD
134750   %=============================================================================
134750   %       LOAD AND STORE IN PHYSICAL MEMORY
134750   %       (64K WORDS ONLY)
134750   %=============================================================================
134750   INTEGER SAVL
134751   PREAD: A:=K1024; X+A; T:=PIOCA; *LDATX                    % DON'T LET CACH FOOL US
134755          A:=K1024; X-A; *LDATX; EXIT
134761   PWRIT: T:=PIOCA; *STATX; EXIT
134764   KREAD: A:=L=:SAVL; CALL PREAD; A SHZ -1; T:=SAVL=:P        % CONVERT TO WORD BASE
134772   RBUS
134774
134774   %       INHIBIT PIOC FROM CACH
134774   %       INPUT A REG POINTES TO FIRST PIOC PAGE
134774
134774   INTEGER OLDL:=77777                       % MAX LOW LIMIT
134775   INTEGER OLDH:=0                           % MAX HIGH LIMIT
134776
134776   SUBR PICAC
134776   INTEGER SAVA
134777   PICAC: A=:SAVA; IF A < OLDL THEN A=:OLDL FI
135004          IF A+PIMPG > OLDH THEN A=:OLDH FI
135011          *TRR 12                           % UCILR
135012          A:=OLDL; *TRR 11                  % LCILR
135014          A:=SAVA; EXIT
135016   RBUS
135020
135020
135020   %       MAKE SURE THAT SOME SEGMENTS ARE FIXED IN ALL PIOC MEMORY
135020
135020   %       INPUT B PIOC DATAFIELD
135020
135020   SUBR FIXOK
135020   INTEGER FPAG,DPAG
135022   FIXOK: T:=HDEV; *IOXT; SHA ZIN SHR 10
135025          A =; PIOCA; *SHA ZIN 6
135027          A*4 =; FPAG; 0=: DPAG
135032          DO WHILE DPAG <= PIMPG                                  % < 77 OR 377
135036            AD:=DCORMSTART; A=:T
135040            DPAG SHZ 2 + FPAG + D =: X; *PAGPR@3 LDATX       % GET SEGMENT NUMBER
135046            A /\377 * 5SEGSIZE + SEGSTART =: X
135052            IF X.FLAG NBIT 5FIX THEN
135055              A:=DEMSV=:B
135057              DPAG =: ZAREG
135061              PNOTFIX; GO PIRET
135063            FI
135063            DPAG + 1 =: DPAG
135066          OD
135067          EXIT
135070   RBUS
```

```
135077
135077    %       FIND OUT IF A SEGMENT ( FOUND IN SEG) IS FIXED WITHIN CURRENT PIOC
135077    %       IF SO UNFIX IT
135077
135077    %       INPUT B PIOC DATAFIELD
135077    %             A SEGMENT NUMBER
135077
135077    SUBR PISEF
135077    INTEGER FPAG
135100    INTEGER DPAG
135101    INTEGER SEG
135102    INTEGER SAVL
135103    PISEF: A=:SEG; T:=HDEV; *IOXT; SHA ZIN SHR 10
135107           A =: PIOCA; *SHA ZIN 6
135111           A*4 =: FPAG; 0=: DPAG
135114           DO WHILE DPAG <= PIMPG                                  % < 77 OR 377
135120             AD:=DCORMSTART; A=:T
135122             DPAG SHZ 2 + FPAG + D =: X; *PAGPR@3 LDATX             % SEGMENT NUMBER
135130             IF A /\377 =  SEG THEN
135134               A * 5SEGSIZE + SEGSTART=:X
135137               IF X.FLAG BIT 5FIX THEN
135142                 A:=L=:SAVL
135144                 A:=SEG; CALL PM116; A:=SAVL=:L                    % UNFIX
135150                 EXIT
135151               FI
135151             FI
135151             DPAG + 1 =: DPAG
135154           OD
135155           EXIT
135156
135156    RBUS
135164
135164    %       CHECK IF YOU OPERATE ON LEGAL SLOT
135164
135164    SUBR CHPO
135164    CHPO:   IF DEMSV.ZAREG >> SLMAX THEN PILSL; GO PIRET; FI       % ILLEGAL SLOT
135173           A + "NDPRO" =:X ; L=:D; CALL PREAD; D=:L
135200           IF A><CURPRG THEN PNOTY; GO PIRET; FI                   % YOU HAVE NOT RESERVED SLOT
135205           EXIT
135206    RBUS
135212
135212    %       CHECK IF PIOC LOADED
135212
135212    SUBR CHRUN
135212    CHRUN: IF PISTT >< 3 THEN                                % MUST BE STARTED AFTER LOAD
135216              PNOTR; GO PIRET
135220           FI
135220           EXIT
135221    RBUS
```

```
135222    %
135222    %=======================================================================
135222    % 39.18      L O A D    S T O P    S T A R T    R E S S L
135222    %             R E L S L    K I C K    W K I C K    U N L O A D
135222    %
135222    %              P I O C    N O N R E S I D E N T (POF OR PIT3)  P A R T
135222    %
135222    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
135222
135222    SUBR PIOCE
135222    INTEGER ARRAY PIENT:=(PIRES,PIREL,PIKIC,PIWKI,PILOA,PIUNL,PISTA,PISTO)
135232    PIOCE: PIENT(A)=:P                                          % GO TO ACTION ROUTINE
135235    RBUS
135236
135236    SUBR PILOA
135236    % LOAD PIOC
135236    % X: LDN
135236    % A: SEGMENT NUMBER
135236    % D: PAGE WITHIN PIOC
135236    INTEGER SEG,PAGE
135240    DOUBLE SEGPA = SEG
135240    PILOA:  T:=HDEV; *IOXT; SHA ZIN SHR 10                      % BANK NUMBER
135243            A*100+DEMSV.ZDREG =: PAGE; X.ZAREG =: SEG
135251          CALL PISEF                                  % UNFIX FIRST IF ALREADY FIXED WITHIN THIS PIOC
135252          SEG BONE 17 =: SEG
135255          AD:=SEGPA; CALL PM160                       % FIX IT
135257          IF A >< 0 THEN
135260            A-;A+40; GO PIRET
135263          FI
135263          0=:PISTT
135264          POK; GO PIRET
135266    RBUS
135273
135273    %       STOP (PANIC)
135273    SUBR  PISTO
135273    PISTO: T:=HDEV+3; A:= 160=:PWCR; *IOXT              % RESET AND HALT
135300         CALL FIXOK
135301         X:=1002; A:=0; CALL PWRIT                    % MARK AS STOPPED IN PIOC
135304         POK; GO PIRET
135306    RBUS
135312
135312
135312
135312    %       START PIOC
135312    %       X: LDN
135312    %       AD: START-ADDR          =0 PIOCOS
135312    %       ON RETURN : IF ERROR A = FIRST NONFIXED PAGE
135312
135312    INTEGER LPOINTERS
135313    INTEGER LKICKP
135314
135314    SUBR PISTA
135314    INTEGER ASAD,DELAY
135316    PISTA: T:=HDEV; *IOXT; SHA ZIN SHR 10
135321           A=:PIOCA; * SHA ZIN 6                      % SAVE PIOC BANK NUMBER
135323           CALL PICAC; CALL FIXOK
135325           X:=1002; A:=0; CALL PWRIT                  % RESET INITFLAG
135330
135330           T:=HDEV+3; A:=60; *IOXT                    % HALT AND RESET
135334           A:=0=:PWCR; *IOXT                          % INITIATE
```

```
135337
135337    %       WAIT FOR PIOC TO GET READY
135337            -3 =: TMR
135341            DO
135341              A:=-500=:DELAY; *MIN DELAY; JMP *-1
135345              X:=1002; CALL PREAD
135347              WHILE A = 0
135350            OD
135351            0 =: TMR
135352            IF A >< PRKEY THEN PPROM; GO PIRET FI
135357    %       GIVE POINTERS TO PIOC DATAFIELD
135357    %       ALL ADDRESSES ARE CONVERTED TO WORD BASE (ND100)
135357
135357            X:=1001; CALL KREAD; A=:LPOINT
135362            A:=PIMPG; X:=LPOINT; *AAX PSIZ
135365            CALL PWRIT; A:=PIOCN; X:=LPOINT; *AAX PINO          % PIOC SIZE (PAGES) TO PIOC
135371            CALL PWRIT                                          % PIOC NUMBER (0,1..) TO PIOC
135372            X:=LPOINT; *AAX PIMBL
135374            CALL PREAD; A=:D; X:=LPOINT; *AAX PIMBH
135400            CALL PREAD; AD=: MBOXH
135402            X:=LPOINT; *AAX RTBL
135404            CALL PREAD; A=:D; X:=LPOINT; *AAX RTBH
135410            CALL PREAD; AD=:RTBOX
135412            X:=LPOINT; *AAX CPUNR
135414            A:=SYSNO; CALL PWRIT                                % ND100 CPU NUMBER
135416            X:=LPOINT; *AAX PNOPC
135420            CALL KREAD; A=:PNBOX                      % PIOC TO NORD MAILBOX
135422            X:=LPOINT; *AAX NPOPC
135424            CALL KREAD; A=:MASTA                      % NORD TO PIOC MAILBOX
135426
135426    %       KICK TABLE POINTERS TO DATAFIELD
135426            X:=LPOINT; *AAX KICKP
135430            CALL KREAD; A =:LKICKP=:PKICK             % ADDR KICKTABLE
135433            X:=LKICKP; *AAX XPTN
135435            CALL KREAD; A=;"PTN"
135437            X:=LKICKP; *AAX XNTP
135441            CALL KREAD; A=;"NTP"
135443            X:=LKICKP; *AAX XPIPR
135445            CALL KREAD; A=:"PIPRO"
135447            X:=LKICKP; *AAX XNDPR
135451            CALL KREAD; A=:"NDPRO"
135453            X:=LKICKP; *AAX XNMAI
135455            CALL KREAD; A=:"PNMAIL"
135457            X:=LKICKP; *AAX XPMAI
135461            CALL KREAD; A=:"NPMAIL"
135463
135463    %       SET START-ADDR
135463
135463            X:=LPOINT; *AAX PIOCO
135465            CALL KREAD; A=:ASAD                       % ADDRESS OF LOCATION HOLDING START ADDRESS
135467            IF DEMSV.ZAREG >< 0 THEN
135472              X:=ASAD; CALL PWRIT
135474            FI
135474            IF DEMSV.ZDREG >< 0 THEN
135477              X:=ASAD+1;  CALL PWRIT
135502            FI
135502
135502
135502    %       SO WE CAN GIVE THE START COMMAND
135502
```

```
135502          X:=MASTA; *AAX NPFUN
135504          A:=MPIOC; CALL PWRIT
135506          X:=MASTA; *AAX NPTIG
135510          A:=TRIG; CALL PWRIT
135512          A:=11; T:=HDEV+3; *IOXT
135516          3 =: PISTT                                  % MARK AS STARTED AFTER LOAD
135520          POK; GO PIRET
135522      RBUS
135540
135540      %       RESERVE SLOT
135540      %       RESERVE SLOT NUMBER GIVEN IN A-REG
135540
135540      SUBR PIRES
135540      PIRES:  CALL CHRUN
135541          IF DEMSV.ZAREG >> SLMAX THEN PILSL; GO PIRET; FI      % ILLEGAL SLOT
135550          A + "NDPRO" =: X; CALL PREAD
135553          IF A >< 0 THEN PSLBS GO PIRET; FI                      % IF FREE THEN
135556          CURPRG; CALL PWRIT                                    % SET IT BUSY BY ME
135560          POK; GO PIRET
135562      RBUS
135570
135570      %       RELEASE SLOT
135570      %       THE SLOT TO BE RELEASED IN A-REG
135570
135570      SUBR PIREL
135570      PIREL:  CALL CHRUN
135571          IF DEMSV.ZAREG >> SLMAX THEN PILSL; GO PIRET; FI      % ILLEGAL SLOT
135600          A + "NDPRO" =: X ; A:=0; CALL PWRIT
135604          POK; GO PIRET
135606      RBUS
135612
135612      %       KICK THROUGH SLOT
135612      %       THE SLOT NUMBER IN A-REG, INFO IN D-REG
135612
135612      SUBR PIKIC
135612      PIKIC:  CALL CHRUN; CALL CHPO
135614          IF DEMSV.ZDREG >< 0 THEN
135617            A := "NPMAIL" + DEMSV.ZAREG =: X; CALL PREAD
135624            IF A >< 0 THEN PFULL; GO PIRET; FI
135627            T:=DEMSV.ZDREG; A := "NPMAIL" + X.ZAREG =: X; T=:A; CALL PWRIT
135636          FI
135636          A := "NTP" + DEMSV.ZAREG =: X; TRIG; CALL PWRIT
135644          A:=PWCR BONE BNDC; T:=HDEV+3; *IOXT
135651          POK; GO PIRET
135653      RBUS
135661
135661      %       WAIT FOR PIOCINF
135661      %       THE SLOT NUMBER IN A-REG, INFO IN D-REG
135661
135661      SUBR PIWKI
135661      PIWKI:  CALL CHRUN; CALL CHPO
135663          A := "PNMAIL" + DEMSV.ZAREG=:X=:D; CALL PREAD
135671          IF A >< 0 THEN                                        % ANY INFO ?
135672            A =: DEMSV.ZDREG                                    % GIVE INFO WORD TO USER
135674            D=:X; A:=0; CALL PWRIT                              % MAILBOX EMTY
135677            POK; GO PIRET
135701          ELSE
135702            PNOMES; GO PIRET                                     % NO MESSAGE FOR YOU
135704          FI
135704      RBUS
```

```
135712
135712    %       UNLOAD PIOC     (UNFIX SEGMENTS FIXED IN PIOC MEMORY)
135712
135712    SUBR PIUNL
135712    INTEGER SEG,FPAG,DPAG
135715    PIUNL: T:=HDEV; *IOXT
135717           A/\ 177400 =: FPAG; 0=: DPAG                    % (BANK*1008*4) INDEX IN MEMORY MAP
135722           T+3; A:=160; *IOXT                              % STOP IT FIRST
135725           DO WHILE DPAG <=PIMPG
135731             AD := DCORMSTART; A=:T
135733             DPAG SHZ 2 + FPAG + D =: X; *PAGPR@3 LDATX
135741             A /\ 377 =: SEG * 5SEGSIZE + SEGSTART =: X
135746             IF X.FLAG BIT 5FIX THEN
135751               A:=SEG; CALL PM116                          % UNFIX
135753             FI
135753             DPAG + 1 =: DPAG
135756           OD
135757           0=:PISTT                                        % NOT LOADED
135760           POK; GO PIRET
135762    RBUS
135771
135771
135771    *"3PITP+3POFP
```

```
"135771    %
135771    SUBR APDRIV
135771    %=============================================================================
135771    %            P D R I V
135771    %
135771    %             P I O C    D R I V E R      (PIT3  OR POF)
135771    %
135771    %=============================================================================
135771    %       DESCRIPTION OF N100-XMSG BOX
135771    DISP 0
135771      DOUBLE NXMSG                % NORMAL XMSG ACTIVATION QUEUE
135771      INTEGER  NXFNC              % STATUS, BIT 1 SET BY ND100 (XMSG FUNC DONE)
135771                                  %       , BIT 3 SET BY PIOC (XMSG FUNC REQUESTED)
135771      DOUBLE NXPAR                % PARAMETER POINTER
135771      INTEGER  NXXTB              % XT BLOCK (GIVEN BY N100 XMSG)
135771      INTEGER NXLB                % LAST LOCAL BANK FOR THIS TASK
135771      INTEGER NXPNU               % PROCESS NUMBER
135771    PSID
135771    DISP 0
135771      DOUBLE NXRTW                % RT ACTIVATION QUEUE
135771      INTEGER NXRTF               % STATUS, BIT 0 SET BY PIOC WHEN RTWAK REQUESTED
135771                                  %         BIT 2 SET BY ND100 WHEN RTWAK COMPLETED
135771    PSID
135771
135771    %-----------------------------------------------------------------------------
135771    % PIWKF:
135771    % WAKE UP ROUTINE, ACTIVATED FROM XMSG
135771    %-----------------------------------------------------------------------------
135771    INTEGER PMAX
135772    PIWKF: 0=:PMAX; AD:=RTBOX; AD SHZ -1; D=:X; A+PIOCA=:T; *LDDTX  % POSSIBLE RT ACTIVATIONS IN PIOC
136001    RTFI:  IF A = 0 AND D = 0 THEN CALL WT12 FI
136005           AD SHZ -1; A+PIOCA=:T: D=:X; *NXXTB@3 LDATX
136012           IF A = L THEN                          % CORRESPONDING XT BLOC
136014             X+10; *NXRTF@3 LDATX
136016             A BONE 2; *NXRTF@3 STATX             % YES, SET RTDONE FOR PIOC
136020             A:=PWCR BONE BNDC; T:=HDEV+3; *IOXT  % WAKE UP PIOC
136025             CALL WT12
136026           FI
136026           T=:D; PMAX+1=:PMAX; IF PMAX > 20 GO FAR WT12 % NOTHING
136036           D=:T; *NXRTW@3 LDDTX
136040           GO RTFI
136041    *)FILL
136042    %-----------------------------------------------------------------------
136042    %  XMMC:
136042    %  HANDLING OF MULTICALL
136042    %  THE ROUTINE CONVERT PARAMETER BLOCKS FROM PIOC 6 WORD FORMAT
136042    %  TO NORD 4 WORD FORMAT AND TRANSLATES ADDRESSES IN READ AND WRITE CALLS
136042    %  IF THE BANK CHANGES FROM LAST USED A DEFINE NEW BANK CALL IS INSERTED
136042    %-----------------------------------------------------------------------
136042    % VARIABES AND POINTERS USED IN DOIT
136042    INTEGER PART=?,PARX=?,DSAV=?,XSAV=?
136042    INTEGER POINTER PFUNC:=FUNC,PASAV:=ASAV,PDSAV:=DSAV,PXSAV:=XSAV
136046    INTEGER NXLBS=?
136046
136046    INTEGER MTSAV,MASAV
136050    DOUBLE MTASAV = MTSAV % INTERMEDIATE STORE FOR T AND A REGISTERS IN A CALL
136050    DOUBLE MDXSAV         % INTERMEDIATE STORE FOR D AND X REGISTERS IN A CALL
136052    DOUBLE MUSSAV         % INTERMEDIATE STORE FOR EV. USERADDRESS
136054    DOUBLE NTASAV         % INTERMEDIATE STORE OF NEXT BLOCK, T & A
136056    DOUBLE NDXSAV         % INTERMEDIATE STORE OF NEXT BLOCK, D & X
```

```
136060   DOUBLE NUSSAV          % INTERMEDIATE STORE OF NEXT BLOCK, EV. USERADDR.
136062   INTEGER INX,INRPA,ONRPA % NUMBER OF XMSG CALL IN AND OUT OF ROUTINE
136065   INTEGER HNOFF,LNOFF
136067   DOUBLE NOFF = HNOFF    % START OF NEXT PARAMETER IN NORD BLOCK (4 WORD BLOCKS)
136067   INTEGER HPOFF,LPOFF
136071   DOUBLE POFF = HPOFF    % START OF NEXT PARAMETER IN PIOC BLOCK (6 WORD BLOCKS
136071
136071   XMMC:  IF PXSAV < 0 GO FAR REEX                    % REEXECUTE TASK'S LAST MULTICALL
136074                                                      % IF < -1 XMSG HANDLES IT
136074          PFUNC BONE XFUSG =: PFUNC                   % ORDER XMSG TO USE USER SEGNO
136077          T := PART; X := PARX; *P4@3 LDDTX           % READ IN PARAMETERBLOCK ADDRESS
136102          AD SHZ -1; T:=D=:PASAV;                     % GET WORDADDR. AND SAVE BANK IN PIOC
136105          IF A >< NXLBS THEN                          % BANK IN PIOC DIFFERS FROM LAST?
136110             T := MCURB; X := LCURB; *NXLB@3 STATX    % SAVE NEW BANK AND OFFSET
136113             T := XFDBK; A + PIOCA; *MON 2XMSG        % SET NEW BANK IN XMSG
136116             GO FAR XPLEV; IF T < 0 GO FAR XPERET
136121          FI
136121          A:=PASAV=:D:=NXLBS
136124
136124          A + PIOCA; X := PXSAV                       % PAR.ADDR IN AD,PAR.NR IN X
136126          AD =: NOFF; AD =: POFF                      % INITIALIZE NEXT BLOCK POINTERS
136130          X =: INRPA; X =: ONRPA; 0 =: INX            % INITIALIZE PARAMETER COUNTS
136133          T := HPOFF; X := LPOFF; *P0@3 LDDTX
136136          AD =: NTASAV; *P2@3 LDDTX                   % NEXT PAR. BLOCK IS FIRST
136140          AD =: NDXSAV; *P4@3 LDDTX
136142          AD =: NUSSAV
136143   LOOP:  INX + 1 =: INX                              % INCREMENT LOOP VARIABLE
136146          AD := NTASAV; AD =: MTASAV                  % DO FOR ALL CALLS IN PAR. BLOCK
136150          AD := NDXSAV; AD =: MDXSAV
136152          AD := NUSSAV; AD =: MUSSAV
136154          IF INX < INRPA THEN                           % LAST CALL IN BLOCK ?
136160             AD := POFF; T := 6; *RADD ST DD           % INCREMENT PIOC BLOCK POINTER
136163             *RADD ADC DA
136164             AD =: POFF; T := A; X := D; *P0@3 LDDTX   % NEW PIOC BLOCK PTR. IN TX
136170             AD =: NTASAV; *P2@3 LDDTX                 % SAVE NEXT T&A FROM OVERWRITE
136172             AD =: NDXSAV; *P4@3 LDDTX                 % SAVE NEXT D&X FROM OVERWRITE
136174             AD =: NUSSAV;                             % SAVE NEXT USERADDR. --"--
136175          FI
136175          AD := MTASAV                               % GET CURR. T & A REGISTERS FROM BLOCK
136176          IF A = 6 OR A = 7 THEN                     % READ OR WRITE CALL ?
136204            MTSAV BONE XFUSG =: MTSAV                % ORDER XMSG TO USE USER SEGNO
136207            AD := MUSSAV; AD SHZ -1; T:=D=:MASAV     % CONVERT TO WORD ADDRESSING T!!
136213            IF A >< NXLBS THEN                       % BANK DIFFERS FROM LAST USED?
136216               T := MCURB; X := LCURB; *NXLB@3 STATX % SAVE NEW BANK AND OFFSET
136221               A + PIOCA; D := A; A := XFDBK         % SET PARS. FOR DEFINE BANK NR.
136224               T := HNOFF; X := LNOFF; *P0@3 STDTX   % INSERT PARAMETERS IN BLOCK
136227               A := T; D := X; T := 4; *RADD ST DD   % INCREMENT NORD BLOCK POINTER
136233               *RADD ADC DA
136234               AD =: NOFF
136235               A := ONRPA; A + 1; A =: ONRPA      % INCREMENT RETURNED PARAMETERCOUNT
136240            FI
136240          FI
136240          T := HNOFF; X := LNOFF                     % START OF NEXT NORD BLOCK
136242          AD := MTASAV; *P0@3 STDTX                  % COPY T & A REGISTERS TO BLOCK
136244          AD := MDXSAV; *P2@3 STDTX                  % COPY D & X REGISTERS TO BLOCK
136246          A := T; D := X; T :=4; *RADD ST DD
136252          *RADD ADC DA                               % INCREMENT NORD BLOCK POINTER (4)
136253          AD =: NOFF;
136254          IF INX < INRPA GO FAR LOOP                 % CONTINUE LOOP ?
136260          X := ONRPA                                 % RETURN NEW PARAMETERCOUNT IN X
```

```
136261          X =: XSAV;                              % PARAMETER COUNT MIGHT BE INCR.
136262          GO FAR XMRT
136263    *)FILL
136270
136270
136270    %-------------------------------------------------------------------------
136270    %       DOIT:
136270    %       THIS POINT IS ACTIVATED WHEN PIOC DOES A XMSG CALL FOR EXECUTION IN ND100
136270    %       THE PARAMETERS ARE FOUND BY A POINTER IN A CHAIN OF BOXES.
136270    %       PARAMETERS ARE SET IN REGISTERS, THE XMSG CALL ARE DONE.
136270    %       REACTIVACTION FROM XMSG MAY ACTIVATE THIS CODE IN TWO DIFFERENT POINTS.
136270    %       1. SKIP RETURN AFTER THE CALL
136270    %       2. IN "XPWAK" IF AN RTENTRY LIKE FUNCTION OCCURS
136270    %-------------------------------------------------------------------------
136270    INTEGER NXLBS
136271    INTEGER PART,PARX
136273    INTEGER FUNC=?,STAT=?,ASAV=?
136273    DOUBLE TDSAV=?,DDSAV=?,ADSAV=?
136273
136273    DOIT:   *NXXTB@3 LDATX
136274            IF A = 0 THEN                           % VIRGIN ?
136275              T:=XFDBK; A:=PIOCA; L:=0; *MON 2XMSG  % ONLY FIRST BANK FOR THE TIME BEEING
136301              GO FAR XPLEV; IF T < 0  GO FAR XPERET
136304              T:=MCURB; *NXPNU@3 LDATX              % PROCESS NUMBER
136306              A=:X; A:=L; A=:PXT(X)                 % XT BLOCK TO PROC TABLE IN DATAFIELD
136311              X:=LCURB; *NXXTB@3 STATX              % XTBLOCK TO BOX
136313              A:="PIWKF"; T:=XFWDF; *MON 2XMSG      % DEFINE RT WAKE UP ADDRESS
136316              GO  FAR XPLEV
136317              IF T < 0  GO FAR XPERET
136321            ELSE
136322              L:=A
136323            FI
136323
136323            T:=MCURB; *NXLB@3 LDATX
136325            A SHZ -1=:NXLBS                         % THIS BLOCK LAST BANK
136327            *NXPAR@3 LDDTX                          % POINTERS TO PARAMETERS
136330            AD SHZ -1; A+PIOCA=:T; D=:X; X=:PARX    % FIRST PARAM (FUNCTION)
136335            T=:PART; *P1@3 LDDTX                    % SAVE PARAM POINTER (TX)
136337            AD=:ADSAV; *P3@3 LDATX
136341            A=:XSAV; *P0@3 LDATX
136343            A=:FUNC
136344            IF A = 6 OR A = 7 THEN                  % READ OR WRITE ?
136352              FUNC BONE XFUSG =: FUNC
136355              T:=PART; *P4@3 LDDTX                  %  GET ADDRESS NO WITHIN PIOC
136357              AD SHZ -1; T:=D=:ASAV                 % BUFFER ADDRESS IN WORD ADDRESS
136362              IF A >< NXLBS THEN                    % NEW BANK WITHIN THIS PIOC ?
136365                T:=MCURB; X:=LCURB; *NXLB@3 STATX
136370                T:=XFDBK; A+PIOCA; *MON 2XMSG       % YES SET NEW BANK FOR XMSG
136373                GO  XPLEV; IF T<0 GO  FAR XPERET
136376              FI
136376            FI
136376            IF FUNC = 44 GO FAR XMMC                % CURRENT CALL XMSG MULTICALL?
136402
136402    XMRT:   IF FUNC = 1 THEN                        % DISCONNECT ?
136406               T:=MCURB; X:=LCURB; A BONE 1; *NXFNC@3  STATX  % YES, WE WILL NOT SEE THIS BLOCK AGAIN
136412               *NXPNU@3 LDATX
136413               A=:X; 0=:PXT(X)
136415            FI
136415    REEX:   X:=XSAV; T:=FUNC; AD:=ADSAV; *MON 2XMSG
136421            GO FAR XPLEV
```

```
136422          GO OVER
136423    *)FILL
136427
136427    INTEGER MXP
136430    INTEGER FUNC,STAT,ASAV,DSAV,XSAV
136435    DOUBLE TDSAV = STAT, DDSAV = DSAV, ADSAV = ASAV
136435
136435    OVER:   T=:STAT; X=:XSAV; AD=:ADSAV
136440
136440    %       XMSG FUNCTION DONE,
136440    %       FIND CORRESPONDING BOX IN QUEUE AND GIVE RETURN PARAMETERS
136440
136440          AD:=MBOXH; AD SHZ -1; A+PIOCA =:T; D=:X; *LDDTX
136446          0=:MXP; GO FIRT
136450    NEXT: T=:D; MXP+1=:MXP
136454          IF MXP > 20 GO FAR XPRET                    % MUST BE SOMETHING WRONG IN PIOC  GIVE IN
136460          IF D > PIOCA  GO FAR XPRET                  % ILLEGAL POINTER
136463          D=:T; *NXXTB@3 LDATX
136465          IF A = L THEN                               % MATCH ?
136467            X=:LCURBOX ; T =: MCURB                   % YES, PIOCA INCLUDED
136471            GO FOUND
136472          FI
136472          *NXMSG@3 LDDTX                              % NEXT POINTER
136473    FIRT: IF A = 0 AND D = 0 GO FAR XPRET             % DID NOT FOUND BOX
136476          AD SHZ -1; D=:X; A+PIOCA=:T; GO NEXT
136503    FOUND: *NXPAR@3 LDDTX                             % PARAMETER POINTER
136504          AD SHZ -1; D=:X; A+PIOCA=:T
136510
136510          AD:=TDSAV; *P0@3 STDTX                      % TA-REG
136512          AD:=DDSAV; *P2@3 STDTX                      % DX-REG
136514
136514          AD:=MDCUR; A=:T; D=:X; *NXFNC@3 LDATX
136520          A BONE 1; *NXFNC@3 STATX                    % XMSG DONE
136522    SPARK: A:=PWCR BONE BNDC; T:=HDEV+3; *IOXT        % WAKE UP PIOC
136527          GO PIXMSG                                   % GET NEW BOX IF ANY
136530    *)FILL
136531
136531    %-------------------------------------------------------------------
136531    %       PROCESS XMSG CALL FORM PIOC, MAIN ENTRY PIONT CALLED FROM
136531    %       DRIVER
136531    %       NB MUST NOT START BEFORE PREVIOUS CALL GAVE A SKIP RETURN
136531    %-------------------------------------------------------------------
136531    PIXMSG:AD:=MBOXH; AD SHZ -1; A+PIOCA=:T; D=:X; *LDDTX      % GET HEADER
136537    TRY:    IF A = 0 AND D = 0 GO XPRET                      % EMPTY QUEUE
136542          AD SHZ -1; A+PIOCA=:T; D=:X
136546          IF A > PIOCA GO XPRET
136551          *NXFNC@3 LDATX
136552          IF A = 0 THEN                               % PIOC REQUEST ?
136553            A BONE 3; *NXFNC@3 STATX                  % YES, SET WORKING
136555            X=:LCURB; T=:MCURB; GO FAR DOIT           % YES, SAVE BOX ADDRESS
136560          FI
136560          *NXMSG@3 LDDTX
136561          GO TRY
136562
136562    XPERE: T:=FUNC; GO FAR FOUND
136564    XPLEV: GO AFXMSG
136565    XPRET: GO AFXMSG
136566    *)FILL
136570    %-------------------------------------------------------------------
136570    %  APDRIV:
```

```
136570    %  PIOC DRIVER MAIN ENTRY POINT
136570    %
136570    %-------------------------------------------------------------------------
136570    INTEGER INN
136571    APDRIV: IF PWCR BIT 5 THEN CALL WT12 FI                    % ACCEPT NO INTERRUPT WHEN NOT STARTED
136575            GO  PIXMSG                                           % PROCESS XMSG CALL
136576    APXMSG:X := PNBOX; CALL PREAD
136600            IF A > 0 THEN                                        % TRIGG TO OPCOM ?
136602                  A:=0; CALL PWRIT
136604                  RTRES=:KPROS                                   % YES, WAKE HIM UP
136606                  IF A >< 0 THEN
136607                    CALL RTACT; CALL WT12
136611                      ELSE
136612                    GO NOINT                                     % NO MONITOR ATTACHED
136613                  FI
136613            FI
136613            X := PKICK; CALL PREAD
136615            IF A > 0 THEN                                        % TRIGG FOR SOMEONE ELSE?
136617                  A:=0; CALL PWRIT
136621                  0=:INN
136622                  FOR INN STEP 1 TO SLMAX DO                       % SEE WHO SHALL HAVE A KICK
136626                    A:="PTN"+INN=:X; CALL PREAD                    %
136632                    IF A = TRIG THEN
136635                      A:=0; CALL PWRIT                           % DTRIGG
136637                      A:="NDPRO"+INN=:X; CALL PREAD; A=:KPROS    % GET PROCESS
136644                      IF A >< 0 THEN
136645                        CALL RTACT; CALL WT12
136647                      FI
136647                    FI
136647                  OD
136653            FI
136653    NOINT:  PWCR BONE BENA=: PWCR; T := HDEV+3; *IOXT            % MUST ENABLE PIOC AFTER DUMMY INTERRUPT
136661            CALL WT12
136662    *)FILL
136666
136666    %       ROUTINE FOR DISCONNECT AT LEVEL 12
136666    %       XT BLOCK IN L-REG
136666    DIS12:  T:=XFDCT; *MON 2XMSG
136670            CALL WT12
136671    RBUS
136672    *"
"136672
136672    *"3PIT
"136672     *XEPT3=*
136672    *XLPT3=XEPT3@-12@12+1777
136672    *XLPT3:
137777  136672    *"
"136672
136672    *"-3PIT
"136672     @EOF
136672
```

** SINTRAN III - VSX/500$ ND-570.146/CXA HILDUR$

NORD SPOOLING SYSTEM/VALHALL III

NORD SPOOLING SYSTEM/VALHALL III