KD11-Z

11/44 POWER FAIL
CKKACA0

AH-F617A-MC

FICHE 1 OF 1

JAN 1980
digital
MADE IN USA

```
.REM    %
```

IDENTIFICATION

PRODUCT CODE:	AC-F615A-MC
PRODUCT NAME:	CKKACA0 11/44 POWER FAIL
DATE CREATED:	1-MAR-79
MAINTAINER:	DIAGNOSTIC ENGINEERING
AUTHORS:	CHUCK ROBINSON

CONTENTS

```
                                1.      ABSTRACT

                                        THIS PROGRAM IS MADE UP OF 11  SUBTESTS  TO  CHECK  OUT  THE
                                        POWER  FAIL ON THE 11/44.  THE 2 MSEC.  POWER DOWN AND POWER
                                        UP TIME IS CHECKED ON EACH POWER FAIL.  INITIALLY POWER FAILS
                                        ARE  TRIED IN ALL PROCESSOR MODES THEN UNDER  ERROR CONDITIONS LIKE
                                         STACK OVERFLOW, TIME OUT, AND ODD ADDRESS AND MEMORY MANAGEMENT ABORTS.
                                         FINALLY A MEMORY VOLATILITY TEST IS RUN ON ALL AVAILABLE MEMORY.


                                2.      REQUIREMENTS

                                2.1     EQUIPMENT

                                        PDP11/44 STANDARD COMPUTER WITH UP TO 1M WORDS OF CORE MEMO-
                                        RY.
                                         THIS PROGRAM WILL WORK WITH AUTOMATIC POWER FAIL HARDWARE
                                         (MANUFACTURING ONLY) OR WITH MANUALLY TRIGGERED POWER
                                         FAILURES FOR EACH TEST.


                                2.2     STORAGE

                                        PROGRAM STORAGE - THE ROUTINES USE MEMORY 0 - 10142


                                2.3     PRELIMINARY PROGRAMS

                                             IT IS ASSUMED THAT CPU,TRAPS,MEMORY MANAGEMENT AND UNIBUS MAP
                                             DIAGNOSTICS HAVE BEEN RUN SUCCESSFULLY.


                                3.      LOADING PROCEDURE

                                        USE STANDARD PROCEDURE FOR ''XXDP'' MEDIA.


                                4.      STARTING PROCEDURE

                                4.1     STARTING ADDRESS

                                         200 IS THE PROGRAM STARTING ADDRESS

                                 4.2    CONTROL SWITCH SETTINGS

                                         SWITCH REGISTER <14> CONTROLS REPEATING OF A  FAILING TEST.
                                         THIS BIT MUST BE SET TO A 1 (AFTER ERROR HALT) EVERY TIME
                                         THE TEST IS TO BE REPEATED.
```

```
147
148
149
```

5. OPERATING PROCEDURE

5.1 MANUAL

START PROGRAM AT ADDRESS 200.
A MESSAGE WILL BE TYPE INDENTIFYING THE NAME OF THE PROGRAM,
THE SIZE OF MEMORY, AND RUNNING INSTRUCTIONS. THE NUMBER OF
EACH TEST WILL BE PRINTED AT THE BEGINNING OF EACH TEST.
MANUALLY POWER DOWN THEN UP AFTER EACH TEST NUMBER APPEARS
ON THE TTY. DO THIS FOR EACH TEST UNTIL THE END OF PASS
MESSAGE IS RECEIVED.

5.2 AUTOMATIC

START THE PROGRAM AT 200,THE PROGRAM SIZES FOR THE PRESENCE OF
AUTOMATIC POWER FAIL HARDWARE.IF FOUND,TESTING WILL BE PROCEED
WITH NO MANUAL INTERVENTION REQUIRED.TEST NUMBERS WILL BE PRINTED
PRIOR TO EACH TEST.

5.3 APT SETUP

THE EXECUTION TIMES PROVIDED IN THE APT SCRIPT THAT FOLLOWS
ARE FOR EXECUTION WITH A 11/44 PROCESSOR, CACHE,
16K CORE MEMORY, AND 300 BAUD.
THE FOLLOWING IS A PROGRAM LOAD FILE USED BY APT;

1. E TABLE 'A' IS USED FOR APT DUMP MODE.

2. E TABLE 'B' IS USED FOR APT QV AND RUN TIME MODES.
   $ENVM=240 SUPRESSES TYPEOUTS TO THE TERMINAL AND
   $SWREG=4000 INHIBITS TEST ITERATIONS

1ST PASS RUN TIME	LONGEST TEST TIME	ADDITIONAL RUN TIME
5	5	0

..... E TABLES .....

	A	B
E-MODE/S-MODE ($ENVM/$ENV)	000/000	240/001
SWITCH REGISTER 1 ($SWREG)	000000	004000
SWITCH REGISTER 2	000000	000000
CPU TYPE/OPTIONS	00/0000	00/0000
MEMORY MAP CODE 1	000/00000000	000/00000000

```
                        5.4     SUBROUTINE ABSTRACT

                                PDWDWN AND POWUP



                                THESE ROUTINES ARE USED TO SAVE AND RESTORE VITAL  REGISTERS
                                AND  TEST  THE TIME ALLOWED FOR POWER FAIL BY THE PROCESSOR.



                        6.      ERRORS



                        6.1     ERROR PRINTOUT



                               THE PROGRAM WILL IDENTIFY THE FAILING TEST AND PROVIDE
                               PERTENANT INFORMATION ON ERROR.



                        6.2     ERROR HALTS

                                AFTER AN ERROR PRINTOUT,THE PROGRAM WILL HALT AND RETURN
                                CONTROL TO THE CONSOLE.
                                IF USING POWER FAIL HARDWARE AND YOU WISH TO STOP THE PROCESSOR
                                YOU SHOULD USE THE HALT SWITCH ON THE FRONT PANEL,THE CONSOLE
                                CTRL P WILL HANG UP TTY.TO GET IN CONSOLE MODE PUT HALT SWITCH
                                ON,TO GET BACK INTO PROGRAM SET THE SWICTH TO RUN.

                                IF AN ERROR OCCURS IN TEST 11 (THE MEMORY VOLITILITY TEST)
                                THE PHYSICAL ADDRESS OF THE BAD MEMORY LOCATION CAN BE
                                FOUND BY USING PAR.OFF AS BITS <21:6> AND ADDING VIR.ADD
                                LOCATION <12:0> TO IT.
                                IF AN ERROR IS DETECTED IN THE POWER DOWN OR POWER UP SUBROUTINE
                                THEN  CONTINUING AFTER THE ERROR HALT,WILL RESTART THE PROGRAM
                                AT ADDRESS 200.
```

```
247
248
249
250
251                             6.3         ERROR RECOVERY
252
253                                          CONTINUE OR RESTART AT 200
254
255
256                             7.          RESTRICTIONS
257
258                                          NONE
259
260
261                             8.          MISCELLANEOUS
262
263
264                             8.1         EXECUTION TIME
265
266                                          5 SEC IF IN AUTOMATIC , N/A IF IN MANUAL
267
268
269                             8.2         STACK POINTER
270
271                                          STACK IS INITALLY SET TO 1100
272
273
274                                 %
```

```
276
289                             .TITLE  CKKACA0 11/44 POWER FAIL
                                ;*COPYRIGHT (C) 1979
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY CHUCK ROBINSON
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
290
291
292                             .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        14            LOOP ON TEST
                                ;*        10            INHIBIT BELL AT END-OF-PASS
293
325                             .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100            ;;FIRST ADDRESS OF THE STACK
001100                          KERSTK= STACK           ;;KERNEL STACK
000700                          SUPSTK= STACK-200       ;;SUPERVISOR STACK
000600                          USESTK= STACK-300       ;;USER STACK
104000                          ERROR=EMT
000004                          SCOPE=IOT
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
177776                          PSW=PS
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
177546                          LKS=    177546          ;;LINE CLOCK (KW11-L) STATUS REGISTER

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE LINE FEED
000015                          CR=     15              ;;CODE CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
000007                          R7=     %7              ;;GENERAL REGISTER
000000                          R10=R0
000001                          R11=R1
000002                          R12=R2
000003                          R13=R3
000004                          R14=R4
```

```
000005                  R15=R5
000006                  SP=     %6              ;;STACK POINTER
000006                  KSP=SP
000006                  SSP=SP
000006                  USP=SP
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                        ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
001000                  SW9=SW09
000400                  SW8=SW08
000200                  SW7=SW07
000100                  SW6=SW06
000040                  SW5=SW05
000020                  SW4=SW04
000010                  SW3=SW03
000004                  SW2=SW02
000002                  SW1=SW01
000001                  SW0=SW00

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
```

```
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
001000                  BIT9=BIT09
000400                  BIT8=BIT08
000200                  BIT7=BIT07
000100                  BIT6=BIT06
000040                  BIT5=BIT05
000020                  BIT4=BIT04
000010                  BIT3=BIT03
000004                  BIT2=BIT02
000002                  BIT1=BIT01
000001                  BIT0=BIT00

                        ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;''T'' BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;''TRAP'' TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000100                  LKVEC=  100             ;;LINE CLOCK (KW11-L) VECTER
000114                  CACHVEC=114             ;;CACHE ERROR INTERRUPT VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
000250                  MMVEC=  250             ;;MEMORY MANAGEMENT VECTOR
                        .SBTTL  CACHE   REGISTER DEFINITIONS


177740                  LOADRS = 177740         ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
177742                  HIADRS = 177742         ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
177744                  MEMERR = 177744         ;;CACHE ERROR REGISTER
177746                  CONTRL = 177746         ;;MEMORY CONTROL REGISTER
177750                  MAINT  = 177750         ;;MEMORY MAINTENENCE REGISTER
177752                  HITMIS = 177752         ;;HIT MISS REGISTER ''1'' IMPLIES HIT IN CACHE

                        .SBTTL  CPU REGISTER DEFINITIONS


177760                  SIZELO = 177760         ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
                                                ;;TO GET TO THE LAST 32 WORDS OF MEMORY
177762                  SIZEHI = 177762         ;;HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
                                                ;;CURRENTLY ALL ZERO
177764                  SYSTID = 177764         ;;SYSTEM ID REGISTER
177766                  CPUERR = 177766         ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
                                                ;;THE TRAP TO ERRVEC (000004)


                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS
```

```
                        :*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

177572                  MMR0=   177572
177574                  MMR1=   177574
177576                  MMR2=   177576
172516                  MMR3=   172516
177572                  SR0=MMR0
177574                  SR1=MMR1
177576                  SR2=MMR2
172516                  SR3=MMR3

                        ;*USER ''I'' PAGE DESCRIPTOR REGISTERS

177600                  UIPDR0= 177600
177602                  UIPDR1= 177602
177604                  UIPDR2= 177604
177606                  UIPDR3= 177606
177610                  UIPDR4= 177610
177612                  UIPDR5= 177612
177614                  UIPDR6= 177614
177616                  UIPDR7= 177616

                        ;*USER ''D'' PAGE DESCRIPTOR REGISTORS

177620                  UDPDR0= 177620
177622                  UDPDR1= 177622
177624                  UDPDR2= 177624
177626                  UDPDR3= 177626
177630                  UDPDR4= 177630
177632                  UDPDR5= 177632
177634                  UDPDR6= 177634
177636                  UDPDR7= 177636

                        ;*USER ''I'' PAGE ADDRESS REGISTERS

177640                  UIPAR0= 177640
177642                  UIPAR1= 177642
177644                  UIPAR2= 177644
177646                  UIPAR3= 177646
177650                  UIPAR4= 177650
177652                  UIPAR5= 177652
177654                  UIPAR6= 177654
177656                  UIPAR7= 177656

                        ;*USER ''D'' PAGE ADDRESS REGISTERS

177660                  UDPAR0= 177660
177662                  UDPAR1= 177662
177664                  UDPAR2= 177664
177666                  UDPAR3= 177666
177670                  UDPAR4= 177670
177672                  UDPAR5= 177672
177674                  UDPAR6= 177674
177676                  UDPAR7= 177676

                        ;*SUPERVISOR ''I'' PAGE DESCRIPTOR REGISTERS
```

```
172200                  SIPDR0= 172200
172202                  SIPDR1= 172202
172204                  SIPDR2= 172204
172206                  SIPDR3= 172206
172210                  SIPDR4= 172210
172212                  SIPDR5= 172212
172214                  SIPDR6= 172214
172216                  SIPDR7= 172216

                        ;*SUPERVISOR 'D'' PAGE DESCRIPTOR REGISTERS

172220                  SDPDR0= 172220
172222                  SDPDR1= 172222
172224                  SDPDR2= 172224
172226                  SDPDR3= 172226
172230                  SDPDR4= 172230
172232                  SDPDR5= 172232
172234                  SDPDR6= 172234
172236                  SDPDR7= 172236

                        ;*SUPERVISOR ''I'' PAGE ADDRESS REGISTERS

172240                  SIPAR0= 172240
172242                  SIPAR1= 172242
172244                  SIPAR2= 172244
172246                  SIPAR3= 172246
172250                  SIPAR4= 172250
172252                  SIPAR5= 172252
172254                  SIPAR6= 172254
172256                  SIPAR7= 172256

                        ;*SUPERVISOR 'D'' PAGE ADDRESS REGISTERS

172260                  SDPAR0= 172260
172262                  SDPAR1= 172262
172264                  SDPAR2= 172264
172266                  SDPAR3= 172266
172270                  SDPAR4= 172270
172272                  SDPAR5= 172272
172274                  SDPAR6= 172274
172276                  SDPAR7= 172276

                        ;*KERNEL ''I'' PAGE DESCRIPTOR REGISTERS

172300                  KIPDR0= 172300
172302                  KIPDR1= 172302
172304                  KIPDR2= 172304
172306                  KIPDR3= 172306
172310                  KIPDR4= 172310
172312                  KIPDR5= 172312
172314                  KIPDR6= 172314
172316                  KIPDR7= 172316

                        ;*KERNEL 'D'' PAGE DESCRIPTOR REGISTERS

172320                  KDPDR0= 172320
```

```
172322                  KDPDR1= 172322
172324                  KDPDR2= 172324
172326                  KDPDR3= 172326
172330                  KDPDR4= 172330
172332                  KDPDR5= 172332
172334                  KDPDR6= 172334
172336                  KDPDR7= 172336

                        ;*KERNEL ''I'' PAGE ADDRESS REGISTERS

172340                  KIPAR0= 172340
172342                  KIPAR1= 172342
172344                  KIPAR2= 172344
172346                  KIPAR3= 172346
172350                  KIPAR4= 172350
172352                  KIPAR5= 172352
172354                  KIPAR6= 172354
172356                  KIPAR7= 172356

                        ;*KERNEL 'D'' PAGE ADDRESS REGISTERS

172360                  KDPAR0= 172360
172362                  KDPAR1= 172362
172364                  KDPAR2= 172364
172366                  KDPAR3= 172366
172370                  KDPAR4= 172370
172372                  KDPAR5= 172372
172374                  KDPAR6= 172374
172376                  KDPAR7= 172376



                        .SBTTL  UNIBUS MAP REGISTER DEFINITIONS


                        ;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
                        ;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'


170200                  MAPL00 = 170200
170202                  MAPH00 = 170202
170204                  MAPL01 = 170204
170206                  MAPH01 = 170206
170210                  MAPL02 = 170210
170212                  MAPH02 = 170212
170214                  MAPL03 = 170214
170216                  MAPH03 = 170216
170220                  MAPL04 = 170220
170222                  MAPH04 = 170222
170224                  MAPL05 = 170224
170226                  MAPH05 = 170226
170230                  MAPL06 = 170230
170232                  MAPH06 = 170232
170234                  MAPL07 = 170234
170236                  MAPH07 = 170236
170240                  MAPL10 = 170240
170242                  MAPH10 = 170242
```

```
170244          MAPL11 = 170244
170246          MAPH11 = 170246
170250          MAPL12 = 170250
170252          MAPH12 = 170252
170254          MAPL13 = 170254
170256          MAPH13 = 170256
170260          MAPL14 = 170260
170262          MAPH14 = 170262
170264          MAPL15 = 170264
170266          MAPH15 = 170266
170270          MAPL16 = 170270
170272          MAPH16 = 170272
170274          MAPL17 = 170274
170276          MAPH17 = 170276
170300          MAPL20 = 170300
170302          MAPH20 = 170302
170304          MAPL21 = 170304
170306          MAPH21 = 170306
170310          MAPL22 = 170310
170312          MAPH22 = 170312
170314          MAPL23 = 170314
170316          MAPH23 = 170316
170320          MAPL24 = 170320
170320          MAPH24 = 170320
170324          MAPL25 = 170324
170326          MAPH25 = 170326
170330          MAPL26 = 170330
170332          MAPH26 = 170332
170334          MAPL27 = 170334
170336          MAPH27 = 170336
170340          MAPL30 = 170340
170342          MAPH30 = 170342
170344          MAPL31 = 170344
170346          MAPH31 = 170346
170350          MAPL32 = 170350
170352          MAPH32 = 170352
170354          MAPL33 = 170354
170356          MAPH33 = 170356
170360          MAPL34 = 170360
170362          MAPH34 = 170362
170364          MAPL35 = 170364
170366          MAPH35 = 170366
170370          MAPL36 = 170370
170372          MAPH36 = 170372
170374          MAPL37 = 170374
170376          MAPH37 = 170376
170200          MAPL0=MAPL00
170202          MAPH0=MAPH00
170204          MAPL1=MAPL01
170206          MAPH1=MAPH01
170210          MAPL2=MAPL02
170212          MAPH2=MAPH02
170214          MAPL3=MAPL03
170216          MAPH3=MAPH03
170220          MAPL4=MAPL04
170222          MAPH4=MAPH04
170224          MAPL5=MAPL05
```

```
                170226                  MAPH5=MAPH05
                170230                  MAPL6=MAPL06
                170232                  MAPH6=MAPH06
                170234                  MAPL7=MAPL07
                170236                  MAPH7=MAPH07
    326                                 .SBTTL  TRAP CATCHER

                000000                          .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                000174                          .=174
    000174      000000              DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
    000176      000000              SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                    .SBTTL  STARTING ADDRESS(ES)
    000200      000137  001424              JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
    327         000500              STACK1=500
    328         000204              $SAVPC=.
    329         000030              .=30
    330 000030  006566              $ERROR
    331 000032  000340              340
    332         000204              .=$SAVPC
    334                             .SBTTL  ACT11 HOOKS

                                    ;;**********************************************************************
                                    ;HOOKS REQUIRED BY ACT11
                000204                  $SVPC=.                 ;SAVE PC
                000046                  .=46
    000046      004220                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                  .=52
    000052      000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
                000204                  .=$SVPC                 ;; RESTORE PC
    335         001100          .=1100
    336                         .SBTTL  APT PARAMETER BLOCK

                                    ;;**********************************************************************
                                    ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                    ;;**********************************************************************
                001100                  .$X=.   ;;SAVE CURRENT LOCATION
                000024                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
    000024      000200                  200     ;;FOR APT START UP
                000044                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
    000044      001100                  $APTHDR ;;POINT TO APT HEADER BLOCK
                001100                  .=.$X   ;;RESET LOCATION COUNTER
                                    ;;**********************************************************************
                                    ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                    ;INTERFACE SPEC.

    001100                          $APTHD:
    001100      000000              $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
    001102      001114              $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
    001104      000005              $TSTM:  .WORD   5       ;;RUN TIM OF LONGEST TEST
    001106      000005              $PASTM: .WORD   5       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
    001110      000000              $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
    001112      000016                      .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
    337                             .SBTTL  APT MAILBOX-ETABLE
```

```
                                ;;*****************************************************************
                                .EVEN
001114                          $MAIL:                          ;;APT MAILBOX
001114  000000                  $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001116  000000                  $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001120  000000                  $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001122  000000                  $PASS:  .WORD   APASS           ;;PASS COUNT
001124  000000                  $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001126  000000                  $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001130  000000                  $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001132  000000                  $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001134                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
001134      000                 $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
001135      000                 $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
001136  000000                  $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
001140  000000                  $USWR:  .WORD   AUSWR           ;;USER SWITCHES
001142  000000                  $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
                                ;*                              BITS 15-11=CPU TYPE
                                ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                                      11/70=06,PDQ=07,Q=10
                                ;*                              BIT 10=REAL TIME CLOCK
                                ;*                              BIT  9=FLOATING POINT PROCESSOR
                                ;*                              BIT  8=MEMORY MANAGEMENT
001144      000                 $MAMS1: .BYTE   AMAMS1          ;;HIGH ADDRESS,M.S. BYTE
001145      000                 $MTYP1: .BYTE   AMTYP1          ;;MEM. TYPE,BLK#1
                                ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                                      900 NSEC CORE=001
                                ;*                                      300 NSEC BIPOLAR=002
                                ;*                                      500 NSEC MOS=003
001146  000000                  $MADR1: .WORD   AMADR1          ;;HIGH ADDRESS,BLK#1
                                ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
001150                          $ETEND:
                                .MEXIT
338                             .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;*****************************************************************
001150  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
001156  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
001164  000403                          BR      $ATYC
001166  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
001174                          $ATYC:
001174  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
001176  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
001200  105767  000206                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
001204  001450                          BEQ     5$              ;;IF NOT:  BR
001206  122767  000001  177720          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
001214  001031                          BNE     3$              ;;IF NOT:  BR
001216  132767  000100  177711          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
001224  001425                          BEQ     3$              ;;IF NOT:  BR
001226  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
001232  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
001240  005767  177650          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
001244  001375                          BNE     1$              ;;IF NOT:  WAIT
001246  010067  177656                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
001252  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
001254  001376                          BNE     2$
001256  166700  177646                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
```

```
001262  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
001264  010067  177642                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
001270  012767  000004  177616          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
001276  000413                          BR      5$
001300  017667  000004  000016  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
001306  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
001314  016746  176456                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
001320  004767  004616                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
001324  000000                  4$:     .WORD   0
001326                          5$:
001326  105767  000062          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
001332  001416                          BEQ     12$             ;;IF NOT:  BR
001334  005767  177574                  TST     $ENV            ;;RUNNING UNDER APT?
001340  001413                          BEQ     12$             ;;IF NOT:  BR
001342  005767  177546          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
001346  001375                          BNE     11$             ;;IF NOT:  WAIT
001350  017667  000004  177540          MOV     @4(SP),$FATAL   ;;GET ERROR #
001356  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
001364  005267  177524                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
001370  105067  000020          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
001374  105067  000013                  CLRB    $LFLG           ;;CLEAR LOG FLAG
001400  105067  000006                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
001404  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
001406  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
001410  000207                          RTS     PC              ;;RETURN
001412     000          $MFLG:  .BYTE   0       ;;MESSG. FLAG
001413     000          $LFLG:  .BYTE   0       ;;LOG FLAG
001414     000          $FFLG:  .BYTE   0       ;;FATAL FLAG
                                .EVEN
            000200      APTSIZE=200
            000001      APTENV=001
            000100      APTSPOOL=100
            000040      APTCSUP=040
339
340 001416  000000      PFAIL: .WORD 0
341 001420  000000      XYZ: .WORD 0
342         177570      SWR=177570
343         177570      DISPLAY=SWR
344 001422  000002      LRTI:   RTI
345
346 001424              BEGIN:
                        .SBTTL  INITIALIZE THE COMMON TAGS
    001424  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                        ;;INITIALIZE A FEW VECTORS
    001430  012737  006514  000034  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
    001436  012737  000340  000036  MOV     #340,@#TRAPVEC+2;LEVEL 7
    001444  005067  177452          CLR     $PASS           ;;CLEAR THE PASS COUNT
    001450  016767  002512  002502  MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                        ;;EQUAL TO A ''-1'', SETUP FOR A SOFTWARE SWITCH REGISTER.
    001456  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
    001462  012737  001516  000004  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
    001470  012767  177570  176072  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
    001476  012767  177570  176064  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
    001504  022777  177777  176056  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
    001512  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                            ;;AND  THE HARDWARE SWR IS NOT = -1
```

```
        001514  000403                          BR      65$                     ;;BRANCH IF NO TIMEOUT
        001516  012716  001524          64$:     MOV     #65$,(SP)               ;;SET UP FOR TRAP RETURN
        001522  000002                          RTI
        001524  012767  000176  176036  65$:     MOV     #SWREG,SWR              ;;POINT TO SOFTWARE SWR
        001532  012767  000174  176030          MOV     #DISPREG,DISPLAY
        001540  012637  000004          66$:    MOV     (SP)+,@#ERRVEC          ;;RESTORE ERROR VECTOR

        001544  005067  177352                  CLR     $PASS                   ;;CLEAR PASS COUNT
        001550  132767  000200  177357          BITB    #APTSIZE,$ENVM          ;;TEST USER SIZE UNDER APT
        001556  001403                          BEQ     67$                     ;;YES,USE NON-APT SWITCH
        001560  012767  001136  176002          MOV     #$SWREG,SWR             ;;NO,USE APT SWITCH REGISTER
        001566                          67$:
    347 001566  012777  004654  003250          MOV     #POWDWN,@DVEC           ;SET UP POWER DOWN VECTOR
    348 001574  012737  004554  000114          MOV     #PARERR,@#CACHVEC       ;SET UP PARITY ERROR VECTOR
    349 001602  012737  000001  004556          MOV     #1,@#PARFLG             ;INITIALIZE THE MULTI PARITY ERROR INDICATOR
    350
    351                                 ;DETERMINE THE SIZE OF MEMORY IN SYSTEM USING SYSMAC SIZE
    352                                 ;ROUTINE
    353
    354 001610  005000                          CLR     R0                      ;MAKE MSB'S = 0
    355 001612  052767  000200  003400          BIS     #BIT07,$KT11            ;TURN ON MEMORY MANAGMENT
    356 001620  004767  003336                  JSR     PC,$SIZE                ;GET PAR VALUE FOR LAST WORD
    357 001624  062767  000037  003634          ADD     #37,$LSTBK              ;FINE LAST ADDRESS OF MEMORY BANK
    358 001632  016701  003630                  MOV     $LSTBK,R1               ;CHECK LAST BANK WITH R1
    359 001636  005201                          INC     R1                      ;SIZE MEMORY
    360 001640  071027  000200                  DIV     #200,R0                 ;FORM THE BLOCK NUMBER OF
    361 001644  005300                          DEC     R0                      ; THE LAST ADDRESS AND
    362 001646  010067  003202                  MOV     R0,LIMIT                ;SAVE IT.
    363 001652  005200                          INC     R0                      ;MULT BY 4
    364 001654  006300                          ASL     R0
    365 001656  006300                          ASL     R0
    366
    367
    368                                 ;THIS ROUTINE SIZE FOR PFAIL HARDWARE,TO AUTOMATIC POWER FAIL THE SYSTEM
    369 001660  012767  177740  177530          MOV     #177740,PFAIL           ;SETUP PFAIL ADDRESS
    370 001666  012767  001710  176110          MOV     #3$,4                   ;SET PC
    371 001674  012767  000340  176104          MOV     #340,6                  ;SET PSW
    372 001702  005777  177510                  TST     @PFAIL                  ;READ PFAIL
    373 001706  000412                          BR      4$                      ;POWER FAIL HARDWARE IS IN SYSTEM
    374 001710  012767  000006  176066  3$:     MOV     #6,4                    ;TRAP RETURN
    375 001716  005067  176064                  CLR     6
    376 001722  005726                          TST     (SP)+                   ;SETUP STACK PC
    377 001724  005726                          TST     (SP)+                   ;SETUP STACK PSW
    378 001726  012767  001420  177462          MOV     #XYZ,PFAIL              ;SET UP DUMMY ADDRESS
    379 001734                          4$:
    380 001734  012767  000006  176042          MOV     #6,4                    ;RESTORE TRAP VEC
    381 001742  005067  176040                  CLR     6
    382
    383                                 ;IDENTIFY THE PROGRAM AND TYPE INSTRUCTIONS
    384
    385 001746  023767  000042  002244          CMP     @#42,$ENDAD
    386 001754  001547                          BEQ     PRETST
    387 001756  104401                          TYPE
    388 001760  007755                          TITLE
    389 001762  104401                          TYPE
    390 001764  010012                          BOOT
    391 001766  010046                          MOV     R0,-(SP)                ;;SAVE R0 FOR TYPEOUT
```

```
                001770  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
392             001772  104401  002000                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                001776  000415                          BR      68$             ;;GET OVER THE ASCIZ
                                                ;;69$:  .ASCIZ  *-K WORDS OF MEMORY EXIST *
                002032                          68$:
393             002032  026727  177360  001420        CMP     PFAIL,#XYZ      ;IS AUTOMATIC POWER FAIL
394             002040  001035                        BNE     APF             ;BRANCH IF YES
395             002042  104401  002050                TYPE    ,71$            ;;TYPE ASCIZ STRING
                002046  000431                        BR      70$             ;;GET OVER THE ASCIZ
                                                ;;71$:  .ASCIZ  <CRLF><LF>* MANUALY INTERRUPT THE POWER AFTER EACH TEST # *
                002132                          70$:
396             002132  000454                        BR      FT
397             002134                          APF:
                002134  104401  002142                TYPE    ,65$            ;;TYPE ASCIZ STRING
                002140  000422                        BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <CRLF><LF>* NO MANUAL INTERVENTION REQUIRED*
                002206                          64$:
398             002206  104401  002214                TYPE    ,67$            ;;TYPE ASCIZ STRING
                002212  000424                        BR      66$             ;;GET OVER THE ASCIZ
                                                ;;67$:  .ASCIZ  <CRLF><LF>* AUTO POWER FAIL TEST HARDWARE EXIST*
                002264                          66$:
399
400             002264                          FT:
                002264  104401  002272                TYPE    ,65$            ;;TYPE ASCIZ STRING
                002270  000401                        BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <CRLF>
                002274                          64$:
401             002274  012767  000001  176616  PRETST: MOV     #1,$TESTN
402                                             ;THIS ROUTINE IS FOR CHECKING 2MS TIMING USING SOB OUT OF SIPARS
403                                             ;BRANCH . OUT OF SIPAR0=1.2US
404                                             ;SOB HAS ONE MORE MICRO STEP THROUGH EQUAL 1.4US
405                                             ;1024 MUL BY 1.4US EQUAL 1.43MS MIN TIME CHECKED FOR ILLUP,ILLDWN
406             002302  012737  012700  172240        MOV     #12700,@#SIPAR0 ;LOAD PAR0 & 1 WITH MOV #1777,R0
407             002310  012737  001777  172242        MOV     #1777,@#SIPAR1
408             002316  012737  077001  172244        MOV     #77001,@#SIPAR2 ;LOAD PAR2 WITH SOB R0,.
409             002324  012737  000207  172246        MOV     #207,@#SIPAR3   ;LOAD PAR3 WITH RTS PC
410                                             ;;*****************************************************************
                                                ;*TEST 1         SIMPLE DOWN/UP TEST (KERNAL)
                                                ;;*****************************************************************
                002332                          TST1:
                002332  104401  002340                TYPE    ,65$            ;;TYPE ASCIZ STRING
                002336  000402                        BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  *1 *
                002344                          64$:
                002344  104401  004234                TYPE,$ENULL
411             002350  005037  177776                CLR     @#PS            ;SET KERNAL MODE
412             002354  012703  002370                MOV     #2$,R3          ;SET POWER UP RETURN
413             002360  012777  000001  177030        MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
414             002366  000001                        WAIT                    ;WAIT FOR POWER FAIL
415             002370  010600                2$:     MOV     SP,R0           ;GET SP
416             002372  022700  001074                CMP     #STACK-4,R0     ;CHECK SP
417             002376  001414                        BEQ     3$              ;SKIP IF OK
418             002400  012767  001074  005300        MOV     #1074,EXP.SP1
419             002406  010067  005276                MOV     R0,REC.SP1
420             002412  012706  000500                MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                              ;FOR ERROR PRINTOUT
                002416  104000                        ERROR
```

```
                                                                ;ERROR
                                                                ;-----
421                                                             ;SP NOT STACK-4
422 002420  007706                      EXP.SP1                 ;PRINT EXPECTED STACK POINTER
423 002422  007710                      REC.SP1                 ;PRINT  RECEIVED STACK POINTER
424 002424  000000                      0
425 002426  000000                      HALT
426 002430  012706  001100      3$:     MOV     #STACK,SP       ;RESET SP
427 002434  013700  001074              MOV     @#STACK-4,R0    ;GET RETURN ADDRESS
428 002440  022700  002370              CMP     #2$,R0          ;CHECK ADDRESS
429 002444  001414                      BEQ     4$              ;SKIP IF OK
430 002446  012767  002370  005226      MOV     #2$,EXP.ADD
431 002454  010067  005224              MOV     R0,REC.ADD
432 002460  012706  000500              MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                ;FOR ERROR PRINTOUT
    002464  104000                      ERROR

                                                                ;ERROR
                                                                ;-----
433                                                             ;ADDRESS ON STACK IS WRONG
434 002466  007702                      EXP.ADD                 ;PRINT EXPECTED ADDRESS
435 002470  007704                      REC.ADD                 ;PRINT RECEIVED ADDRESS
436 002472  000000                      0
437 002474  000000                      HALT
438 002476  013700  001076      4$:     MOV     @#STACK-2,R0    ;GET OLD PS
439 002502  022700  000000              CMP     #0,R0           ;CHECK OLD PS
440 002506  001414                      BEQ     5$              ;SKIP IF OK
441 002510  012767  000000  005174      MOV     #0,EXP.PSW
442 002516  010067  005172              MOV     R0,REC.PSW
443 002522  012706  000500              MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                ;FOR ERROR PRINTOUT
    002526  104000                      ERROR

                                                                ;ERROR
                                                                ;-----
444                                                             ;OLD PS IS WRONG
445 002530  007712                      EXP.PSW                 ;PRINT EXPECTED PS
446 002532  007714                      REC.PSW                 ;PRINT RECEIVED PS
447 002534  000000                      0
448 002536  000000                      HALT
449 002540                      5$:
    002540  032737  040000  177570      BIT     #SW14,@#SWR     ;LOOP ON TEST?
    002546  001271                      BNE     TST1            ;LOOP TO TST1
450
451
452
```

```
                                        ;
                                        ;TEST ROUTINE TO CHECK RE-START CAPABILITY
                                        ;USING THE BR. INSTRUCTION
                                        ;OPERATOR MUST SET HALT SWITCH TO ENABLE POSITION
                                        ;
002550  005267  176344                  INC     $TESTN
                                        ;;*****************************************************************
                                        ;*TEST 2        POWER FAIL WITH BR. INSTRUCTION
                                        ;;*****************************************************************
                                        TST2:
002554                                  TYPE    ,65$            ;;TYPE ASCIZ STRING
002554  104401  002562                  BR      64$             ;;GET OVER THE ASCIZ
002560  000402                  ;;65$:  .ASCIZ  *2 *
                                        64$:
002566                                  TYPE,$ENULL
002566  104401  004234
002572  012706  001100                  MOV     #STACK,SP       ;SET UP STACK
002576  012767  000357  175172          MOV     #357,PS                 ;SET UP CONDITION CODES
002604  012703  002620                  MOV     #2$,R3                  ;SET POWER UP RETURN
002610  012777  000001  176600          MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
002616  000777                  1$:     BR      .                       ;WAIT FOR POWER FAIL
002620  010600                  2$:     MOV     SP,R0                   ;GET SP
002622  012706  001100                  MOV     #STACK,SP               ;SET STACK
002626  022700  001074                  CMP     #STACK-4,R0             ;CHECK STACK POINTER
002632  001414                          BEQ     3$                      ;SKIP IF OK
002634  012767  001074  005044          MOV     #1074,EXP.SP1
002642  010067  005042                  MOV     R0,REC.SP1
002646  012706  000500                  MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                ;FOR ERROR PRINTOUT
002652  104000                          ERROR

                                                                        ;ERROR
                                                                        ;-----
                                                                        ;STACK POINTER
                                        EXP.SP1                         ;PRINT RECEIVED SP
                                        REC.SP1                         ;PRINT EXPECTED SP
                                        0
                                        HALT
002664  013700  001076          3$:     MOV     @#STACK-2,R0    ;GET OLD PS
002670  022700  000341                  CMP     #341,R0         ;CHECK OLD PS
002674  001414                          BEQ     4$              ;SKIP IF OK
002676  012767  000341  005006          MOV     #341,EXP.PSW
002704  010067  005004                  MOV     R0,REC.PSW
002710  012706  000500                  MOV     #STACK1,SP              ;SETUP RESERVED STACK POINTER
                                                                        ;FOR ERROR PRINTOUT
002714  104000                          ERROR

                                                                        ;ERROR
                                                                        ;-----
                                                                        ;OLD PS IS WRONG
002716  007712                          EXP.PSW                         ;PRINT EXPECTED OLD PS
002720  007714                          REC.PSW                         ;PRINT RECEIVED OLD PS
002722  000000                          0
002724  000000                          HALT
002726  013700  001074          4$:     MOV     @#STACK-4,R0    ;GET RETURN ADDRESS
002732  022700  002616                  CMP     #1$,R0
002736  001414                          BEQ     5$              ;SKIP IF OK
002740  012767  002616  004734          MOV     #1$,EXP.ADD
```

```
                                                                J 2
    493 002746  010067  004732                  MOV     R0,REC.ADD
    494 002752  012706  000500                  MOV     #STACK1,SP                  ;SETUP RESERVED STACK POINTER
                                                                                    ;FOR ERROR PRINTOUT
        002756  104000                          ERROR

                                                                                    ;ERROR
                                                                                    ;-----
    495                                                                             ;ADDRESS ON STACK IS WRONG
    496 002760  007702                          EXP.ADD                             ;PRINT EXPECTED ADDRESS
    497 002762  007704                          REC.ADD                             ;PRINT RECEIVE ADDRESS
    498 002764  000000                          0
    499 002766  000000                          HALT
    500 002770                          5$:
        002770  032737  040000  177570          BIT     #SW14,@#SWR     ;LOOP ON TEST?
        002776  001266                          BNE     TST2            ;LOOP TO TST2
    501
    502                                 ;TEST ROUTINE TO CHECK RESTART CAPABILITY
    503                                 ;USING THE EMULATOR TRAP FOR A WAIT
    504                                 ;OPERATOR MUST SET HALT SWITCH TO ENABLE POSITION
    505                                 ;
    506 003000  005267  176114                  INC     $TESTN
    507                                 ;;*****************************************************************
                                        ;*TEST 3         POWER FAIL WITH EMT TRAP
                                        ;;*****************************************************************
        003004                          TST3:
        003004  104401  003012                  TYPE    ,65$            ;;TYPE ASCIZ STRING
        003010  000402                          BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  *3 *
        003016                          64$:
        003016  104401  004234                  TYPE,$ENULL
    508 003022  005037  177776                  CLR     @#PS            ;SET KERNAL MODE
    509 003026  012706  001100                  MOV     #STACK,SP               ;SET UP STACK
    510 003032  012703  003064                  MOV     #2$,R3                  ;SET POWER UP RETURN
    511 003036  012767  001422  174764          MOV     #LRTI,EMTVEC            ;SET UP RETURN FROM  EMT TRAP VECTOR
    512 003044  012767  000005  174760          MOV     #5,EMTVEC+2             ;SET PSW ON POWER FAIL
    513 003052  012777  000001  176336          MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
    514 003060  104000                  3$:     EMT                                     ;WAIT FOR POWER FAIL
    515 003062  000776                          BR      3$
    516 003064  010600                  2$:     MOV     SP,R0                   ;GET SP
    517 003066  012706  001100                  MOV     #STACK,SP
    518 003072  012767  006566  174730          MOV     #$ERROR,30              ;RESET EMT VECTOR
    519 003100  022700  001074                  CMP     #STACK-4,R0             ;WAS STACK PUSHED ONLY TWICE
    520 003104  001461                          BEQ     6$
    521 003106  022700  001070                  CMP     #STACK-10,R0            ;WAS STACK PUSHED 4 TIMES
    522 003112  001414                          BEQ     4$
    523 003114  012767  001070  004564          MOV     #1070,EXP.SP1
    524 003122  010067  004562                  MOV     R0,REC.SP1
    525 003126  012706  000500                  MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
        003132  104000                          ERROR

                                                                                    ;ERROR
                                                                                    ;-----
    526                                                                             ;WAS STACK PUSHED 4 TIMES
    527 003134  007706                          EXP.SP1                             ;PRINT EXPECTED STACK POINTER
    528 003136  007710                          REC.SP1                             ;PRINT RECEIVED STACK POINTER
    529 003140  000000                          0
```

```
                                        HALT
003144  013700  001070          4$:   MOV     @#STACK-10,R0
003150  022700  001422                CMP     #LRTI,R0        ;DOES STACK CONTAIN CORRECT INFO
003154  001414                        BEQ     5$
003156  012767  001422  004516        MOV     #LRTI,EXP.ADD
003164  010067  004514                MOV     R0,REC.ADD
003170  012706  000500                MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                              ;FOR ERROR PRINTOUT
003174  104000                        ERROR

                                                                      ;ERROR
                                                                      ;-----
                                              ;IS STACK PUSHED MORE THAN 2 OR LESS THAN 4
003176  007702                        EXP.ADD                         ;PRINT EXPECTED ADDRESS
003200  007704                        REC.ADD                         ;PRINT RECEIVED ADDRESS
003202  000000                        0
003204  000000                        HALT
003206  013700  001072          5$:   MOV     @#STACK-6,R0
003212  022700  000005                CMP     #5,R0           ;DOES STACK CONTAIN CORRECT INFO
003216  001414                        BEQ     6$
003220  012767  000005  004464        MOV     #5,EXP.PSW
003226  000000  007714                R0,REC.PSW
003232  012706  000500                MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                              ;FOR ERROR PRINTOUT
003236  104000                        ERROR

                                                                      ;ERROR
                                                                      ;-----
                                                                      ;STACK CONTIAN WRONG PSW
003240  007712                        EXP.PSW                         ;PRINT EXPECTED PSW
003242  007714                        REC.PSW                         ;PRINT RECEIVED PSW
003244  000000                        0
003246  000000                        HALT
003250                          6$:
003250  032737  040000  177570        BIT     #SW14,@#SWR     ;LOOP ON TEST?
003256  001252                        BNE     TST3            ;LOOP TO TST3


003260  005267  175634                INC     $TESTN
                        ;;*******************************************************************
                        ;*TEST 4         POWER FAIL WITH ODD ADDRESS
                        ;;*******************************************************************
003264                  TST4:
003264  104401  003272                TYPE    ,65$            ;;TYPE ASCIZ STRING
003270  000402                        BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  *4 *
003276                  64$:
003276  104401  004234                TYPE,$ENULL
003302  005037  177776                CLR     @#PS            ;SET KERNAL MODE
003306  012737  003326  000004        MOV     #3$,@#ERRVEC    ;SET TRAP VECTOR
003314  012703  003346                MOV     #1$,R3          ;SET RETURN ADDRESS FOR POWER FAIL
003320  012777  000001  176070        MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
003326  012706  001100          3$:   MOV     #STACK,SP       ;RESET STACK
003332  005737  000003                TST     @#3             ;CAUSE ODD ADDRESS TRAP
003336  012706  000500                MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                              ;FOR ERROR PRINTOUT
```

```
        003342  104000                          ERROR

                                                                        ;ERROR
                                                                        ;-----
    566 003344  000000                          HALT                    ;ODD ADDRESS TRAP FAILED TO OCCUR
    567 003346  012737  000006  000004  1$:     MOV     #6,@#ERRVEC     ;RESET 4
    568 003354  032737  040000  177570          BIT     #SW14,@#SWR     ;LOOP ON TEST?
        003362  001340                          BNE     TST4            ;LOOP TO TST4
    569
    570
    571 003364  005267  175530                  INC     $TESTN
    572                             ;;**************************************************************
                                    ;*TEST 5         POWER FAIL WITH TIME OUT (KERNAL)
                                    ;;**************************************************************
        003370                      TST5:
        003370  104401  003376                  TYPE    ,65$            ;;TYPE ASCIZ STRING
        003374  000402                          BR      64$             ;;GET OVER THE ASCIZ
                                    ;;65$:  .ASCIZ  *5 *
        003402                      64$:
        003402  104401  004234                  TYPE,$ENULL
    573 003406  012737  003426  000004          MOV     #3$,@#ERRVEC    ;SET TRAP VECTOR
    574 003414  012703  003452                  MOV     #1$,R3          ;SET UP RETURN ADDRESS FOR POWER FAIL
    575 003420  012777  000001  175770          MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
    576 003426  012706  001100          3$:     MOV     #STACK,SP       ;SET STACK
    577 003432  005037  177776                  CLR     @#PS            ;SET KERNAL MODE
    578 003436  010037  160000                  MOV     R0,@#160000     ;CAUSE A TIMEOUT
    579 003442  012706  000500                  MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
        003446  104000                          ERROR

                                                                        ;ERROR
                                                                        ;-----
    580 003450  000000                          HALT                    ;TIME OUT FAILED TO OCCUR
    581 003452  012706  001100          1$:     MOV     #STACK,SP       ;SET STACK
    582 003456  012737  000006  000004          MOV     #6,@#ERRVEC     ;RESET 4
    583 003464  032737  040000  177570          BIT     #SW14,@#SWR     ;LOOP ON TEST?
        003472  001336                          BNE     TST5            ;LOOP TO TST5
```

```
                                        INC     $TESTN
                        ;;*************************************************************
                        ;*TEST 6         POWER FAIL IN THE STACK OVERFLOW (KERNAL)
                        ;;*************************************************************
003500                  TST6:
003500  104401  003506          TYPE    ,65$            ;;TYPE ASCIZ STRING
003504  000402                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  *6 *
003512                  64$:
003512  104401  004234          TYPE,$ENULL
003516  005037  177776          CLR     @#PS            ;SET KERNAL MODE
003522  005067  001324          CLR     FLAG            ;CLEAR THE FLAG
003526  012737  003566  000004  MOV     #2$,@#ERRVEC    ;SET SICK TPAP ADDRESS
003534  012706  000400          MOV     #400,SP         ;SET STACK TO YELLOW ZONE
003540  012703  003562          MOV     #1$,R3          ;SET RETURN ADDRESS FOR POWER FAIL
003544  012777  000001  175644  MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
003552  000001                  WAIT                    ;WAIT FOR  THE POWER FAIL
003554  012706  000500          MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                        ;FOR ERROR PRINTOUT
003560  104000                  ERROR

                                                                ;ERROR
                                                                ;-----
003562  000000          1$:     HALT
003564  000407                  BR      4$              ;SKIP SP CHECK
003566  012737  000006  000004  2$:     MOV     #6,@#ERRVEC     ;RESET 4
003574  012703  003604          5$:     MOV     #4$,R3          ;SET RETURN
003600  000002                  RTI                     ;GO TO THE POWER FAIL ROUTINE
003602  000000                  HALT
003604                  4$:
003604  032737  040000  177570  BIT     #SW14,@#SWR     ;LOOP ON TEST?
003612  001332                  BNE     TST6            ;LOOP TO TST6

                                        INC     $TESTN
                        ;;*************************************************************
                        ;*TEST 7         POWER FAIL WITH RESETS
                        ;;*************************************************************
003620                  TST7:
003620  104401  003626          TYPE    ,65$            ;;TYPE ASCIZ STRING
003624  000402                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  *7 *
003632                  64$:
003632  104401  004234          TYPE,$ENULL
003636  005037  177776          CLR     @#PS            ;SET KERNAL MODE
003642  012703  003700          MOV     #1$,R3          ;SET RETURN ADDRESS
003646  012706  001100          MOV     #STACK,SP       ;RESET STACK
003652  012777  000001  175536  MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
003660  000005          3$:     RESET                   ;RESETS
003662  000005                  RESET                   ;TO WAIT
003664  000005                  RESET                   ;IN
003666  000774                  BR      3$              ;LOOP
003670  012706  000500          MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                        ;FOR ERROR PRINTOUT
003674  104000                  ERROR

                                                                ;ERROR
```

```
                                                                    ;------
615                                                                 ;RESET LOOP FAILED
616 003676  000000                          HALT
617 003700  012706  001100          1$:     MOV     #STACK,SP       ;RESET STACK
618 003704  032737  040000  177570          BIT     #SW14,@#SWR     ;LOOP ON TEST?
    003712  001342                          BNE     TST7            ;LOOP TO TST7
```

```
                                        INC     $TESTN
                        ;;*****************************************************************
                        ;*TEST 10        MEMORY MANAGEMENT ABORT TEST
                        ;;*****************************************************************
    003720              TST10:
    003720  104401  003726          TYPE    ,65$            ;;TYPE ASCIZ STRING
    003724  000402                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  *10 *
    003732              64$:
    003732  104401  004234          TYPE,$ENULL
626 003736  005037  177776          CLR     @#PS            ;SET KERNAL MODE
627 003742  012737  004016  000004  MOV     #4$,@#ERRVEC    ;SET FOR TIMEOUT
628 003750  004767  001104          JSR     PC,MAP  ;MAP THE WORLD
629 003754  012737  004000  000250  MOV     #3$,@#MMVEC     ;SET MEMORY MANAGEMENT VECTOR
630 003762  012703  004020          MOV     #1$,R3          ;LOAD PF RETURN
631 003766  012777  000001  175422  MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
632 003774  005237  177572          INC     @#MMR0          ;TURN MEMORY MANAGEMENT ON
633 004000  012706  001100      3$: MOV     #STACK,SP       ;ZAP STACK
634 004004  005237  140000          INC     @#140000        ;ACCESS VIOLATION
635 004010  012706  000500          MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                            ;FOR ERROR PRINTOUT
    004014  104000          ERROR

                                                                    ;ERROR
                                                                    ;-----
636                                                                 ;NO VIOLATION OR TRAP TO 4
637 004016  000000      4$: HALT
638
639 004020  005037  177572      1$: CLR     @#MMR0          ;TURN OFF MEMORY MANAGEMENT
640 004024  012706  001100      2$: MOV     #STACK,SP       ;MAKE A NEW STACK
641 004030  012737  000006  000004  MOV     #6,@#ERRVEC     ;RESET 4
642 004036  032737  040000  177570  BIT     #SW14,@#SWR     ;LOOP ON TEST?
    004044  001325              BNE     TST10           ;LOOP TO TST10
643
644 004046  005267  175046          INC     $TESTN
645                     ;;*****************************************************************
                        ;*TEST 11        MEMORY VOLATILITY TEST
                        ;;*****************************************************************
    004052  000240      TST11:  NOP
646 004054  005037  177776          CLR     @#PS            ;SET KERNAL MODE
647 004060  004767  000170      4$: JSR     PC,LOAD         ;LOAD ALL MEMORY WITH 52525
648 004064  012703  004122          MOV     #1$,R3          ;POWER FAIL RETURN ADDRESS
649 004070  104401  004076          TYPE    ,65$            ;;TYPE ASCIZ STRING
    004074  000402                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  *11 *
    004102              64$:
    004102  104401  004234          TYPE,$ENULL
650 004106  004767  000246      2$: JSR     PC,CHECK        ;CHECK FOR THE 52525
651 004112  012777  000001  175276  MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
652 004120  000772              BR      2$              ;LOOP FOR EVER OR POWER FAIL
653 004122  012706  001100      1$: MOV     #STACK,SP       ;ZAP THE STACK
654 004126  004767  000226          JSR     PC,CHECK        ;CHECK ALL MEMORY
655 004132  032737  040000  177570  BIT     #SW14,@#SWR     ;LOOP ON TEST?
```

```
004140  001344                  BNE     TST11           ;LOOP TO TST11
```

```
    657                                 .SBTTL  END OF PASS ROUTINE

                                        ;;**********************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO FT

004142                                  $EOP:
004142  000240                                  NOP
004144  005267  174752                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
004150  042767  100000  174744                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
004156  005327                                  DEC     (PC)+           ;;LOOP?
004160  000001                          $EOPCT: .WORD   1
004162  003022                                  BGT     $DOAGN          ;;YES
004164  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
004166  000001                          $ENDCT: .WORD   1
004170  004160                                  $EOPCT
004172  104401  004237                          TYPE    ,$ENDMG         ;;TYPE 'END PASS #'
004176  016746  174720                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
004202  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
004204  104401  004234                          TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
004210  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
004214  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
004216  000005                                  RESET                   ;;CLEAR THE WORLD
004220  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
004222  000240                                  NOP                     ;;SAVE ROOM
004224  000240                                  NOP                     ;;FOR
004226  000240                                  NOP                     ;;ACT11
004230                                  $DOAGN:
004230  000137                                  JMP     @(PC)+          ;;RETURN
004232  002264                          $RTNAD: .WORD   FT
004234     377     377     000         $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
004237     015     012     105         $ENDMG: .ASCIZ  <15><12>/END PASS #/
004242     116     104     040
004245     120     101     123
004250     123     040     043
004253     000
```

```
659
660                                             .SBTTL  LOAD MEMORY WITH DATA PATTERN
661 004254  004767  000600                      LOAD:   JSR     PC,MAP          ;SET UP MEMORY MANAGEMENT REGISTERS
662 004260  016704  000570                              MOV     LIMIT,R4        ;GET BANK COUNT
663 004264  016705  000566                              MOV     DATA,R5         ;GET THE DATA PATTERN
664 004270  012737  004320  000250                      MOV     #2$,@#MMVEC     ;SET UP FOR MEMORY MANAGEMENT ABORTS
665 004276  012700  010170                              MOV     #END,R0         ;FORM ADDRESS OF WHERE TO START WRITING THE DATA
666 004302  062700  120000                              ADD     #120000,R0      ;USE KIPAR5 FOR ADDRESSING
667 004306  012737  000001  177572                      MOV     #1,@#MMR0       ;TURN ON MEMORY MANAGEMENT
668 004314  010520                      1$:     MOV     R5,(R0)+        ;WRITE THE DATA
669 004316  000776                              BR      1$
670 004320  005304                      2$:     DEC     R4              ;LAST BANK WRITTEN?
671 004322  100411                              BMI     3$              ;BRANCH IF YES
672 004324  012700  120000                      MOV     #120000,R0      ;RESET ADDRESSING REGISTER
673 004330  062737  000200  172352              ADD     #200,@#KIPAR5   ;SIZE NEXT 4K BANK
674 004336  012737  000001  177572              MOV     #1,@#MMR0       ;CLEAR MEMORY MANAGEMENT ERRORS
675 004344  000002                              RTI                     ;RETURN TO CONTINUE WRITING
676 004346  005037  177572                      3$:     CLR     @#MMR0          ;TURN OFF MEMORY MANAGEMENT
677 004352  062706  000004                              ADD     #4,SP           ;CLEAN UP THE STACK
678 004356  000207                              RTS     PC              ;RETURN TO CALLER
```

```
                                        .SBTTL  CHECK MEMORY FOR CORRECT DATA PATTERN
                                        
004360  004767  000474          CHECK:  JSR     PC,MAP          ;SET UP MEMORY MANAGEMENT REGISTERS
004364  016704  000464                  MOV     LIMIT,R4        ;GET BANK COUNT
004370  016705  000462                  MOV     DATA,R5         ;GET DATA PATTERN
004374  012737  004514  000250          MOV     #2$,@#MMVEC     ;SET UP FOR MEMORY MANAGEMENT ABORTS
004402  012700  010170                  MOV     #END,R0         ;ADDRESS OF WHERE DATA PATTERN STARTS
004406  062700  120000                  ADD     #120000,R0      ;USE KIPAR5 FOR ADDRESSING
004412  012737  000001  177572          MOV     #1,@#MMR0       ;TURN ON MEM. MANAGEMENT
004420  012001                  1$:     MOV     (R0)+,R1        ;READ THE DATA
004422  020501                          CMP     R5,R1           ;IS IT GOOD?
004424  001775                          BEQ     1$              ;BRANCH IF YES
004426  010067  003264                  MOV     R0,VIR.ADD
004432  162767  000002  003256          SUB     #2,VIR.ADD
004440  042767  160000  003250          BIC     #160000,VIR.ADD
004446  013767  172352  003244          MOV     @#KIPAR5,PAR.OFF
004454  010567  003216                  MOV     R5,EXPDAT
004460  010167  003214                  MOV     R1,RECDAT
004464  012706  000500                  MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                ;FOR ERROR PRINTOUT
004470  104000                          ERROR

                                                                                ;ERROR
                                                                                ;-----
004472  007716                          VIR.ADD                 ;PRINT VIRTUAL ADDRESS
004474  007720                          PAR.OFF                 ;PRINT PDR OFFSET
004476  007676                          EXPDAT                  ;PRINT EXPECTED DATA
004500  007700                          RECDAT                  ;PRINT RECEIVED DATA
004502  000000                          0
004504  000000                          HALT
004506  010560  177776                  MOV     R5,-2(R0)       ;WRITE GOOD DATA
004512  000742                          BR      1$              ;GO READ THE NEXT WORD
004514  005304                  2$:     DEC     R4              ;LAST BANK COMPARED?
004516  100411                          BMI     3$              ;BRANCH IF YES
004520  012700  120000                  MOV     #120000,R0      ;RESET ADDRESSING REGISTER
004524  062737  000200  172352          ADD     #200,@#KIPAR5   ;SIZE NEXT 4K BANK
004532  012737  000001  177572          MOV     #1,@#MMR0       ;CLEAR MEMORY MANAGEMENT ERRORS
004540  000002                          RTI                     ;CONTINUE TESTING
004542  005037  177572          3$:     CLR     @#MMR0          ;TURN OFF MEM. MANAGEMENT
004546  062706  000004                  ADD     #4,SP           ;CLEAN UP THE STACK
004552  000207                          RTS     PC              ;RETURN FOR CALL
```

```
                                        .SBTTL  PARITY ERROR HANDLER

004554  005327                  PARERR: DEC     (PC)+           ;FIRST TIME IN?
004556  000001                  PARFLG: .WORD   1
004560  002002                          BGE     2$              ;BRANCH IF YES
004562  000000                  1$:     HALT                    ;NO---ANOTHER PARITY ERROR OCCURRED WHILE
004564  000776                          BR      1$              ;PROCESSING THE FIRST ONE.
004566  032767  000077  173146  2$:     BIT     #77,HIADRS      ;MSB'S OF PARITY ERROR ADDRESS=0?
004574  001006                          BNE     4$              ;BRANCH IF NO
004576  023727  177740  010170          CMP     @#LOADRS,#END   ;LSB'S ABOVE THE PROGRAM?
004604  101002                          BHI     4$              ;BRANCH IF YES
004606  000000                  3$:     HALT                    ;PARITY ERROR IS IN THE PROGRAM
004610  000776                          BR      3$
004612  000000                  4$:     HALT                    ;DATA PARITY ERROR OCCURRED
004614  005737  177744                  TST     @#MEMERR        ;DID AN ABORT OCCUR?
004620  100405                          BMI     5$              ;BRANCH IF YES
004622  162700  000002                  SUB     #2,R0           ;BACKUP BY ONE WORD
004626  012705  000002                  MOV     #BIT01,R5
004632  074500                          XOR     R5,R0           ;
004634  010320                  5$:     MOV     R3,(R0)+        ;REWRITE THE BAD DATA
004636  013737  177744  177744          MOV     @#MEMERR,@#MEMERR ;CLEAR ERROR INDICATORS
004644  012767  000001  177704          MOV     #1,PARFLG       ;INITIALIZE PARITY ERROR FLAG
004652  000002                          RTI                     ;CONTINUE TESTING
```

```
                                        .SBTTL  POWER FAIL ROUTINE

004654  012767  177777  000170  POWDWN: MOV     #-1,FLAG        ;FIRST INSTRUCTION FLAG
004662  005067  000164                  CLR     FLAG            ;NOW CLEAR IT
004666  012777  005012  000144          MOV     #ILLUP,@UVEC    ;IF TOO FAST
004674  011667  000136                  MOV     (SP),ERRAD      ;SET THE ERROR ADDRESS
004700  022706  000400                  CMP     #400,SP         ;YELLOW OR RED?
004704  100402                          BMI     1$              ;NO
004706  012706  001100                  MOV     #STACK,SP       ;SET EMERGENCY STACK
004712  010046                  1$:     MOV     R0,-(6)         ;PUT
004714  010146                          MOV     R1,-(6)         ;THE
004716  010246                          MOV     R2,-(6)         ;REGISTERS
004720  010346                          MOV     R3,-(6)         ;ON
004722  010446                          MOV     R4,-(6)         ;THE
004724  010546                          MOV     R5,-(6)         ;STACK
004726  010667  000116                  MOV     SP,SAV6         ;SAVE THE STACK POINTER
004732  004737  172240                  JSR     PC,@#SIPAR0     ;SET TIME FACTOR
004736  012777  004750  000074          MOV     #POWUP,@UVEC    ;RESET THE UP VECTOR
004744  000000                          HALT                    ;WAIT FOR POWER DOWN
004746  000240                          NOP

004750  012777  005024  000066  POWUP:  MOV     #ILLDWN,@DVEC   ;SET TOO FAST DOWN VECTOR
004756  016706  000066                  MOV     SAV6,SP         ;RESET SP
004762  004737  172240                  JSR     PC,@#SIPAR0     ;SET TIME FACTOR
004766  012605                          MOV     (6)+,R5         ;TAKE
004770  012604                          MOV     (6)+,R4         ;THE
004772  012603                          MOV     (6)+,R3         ;REGISTERS
004774  012602                          MOV     (6)+,R2         ;FROM
004776  012601                          MOV     (6)+,R1         ;THE
005000  012600                          MOV     (6)+,R0         ;STACK
005002  012777  004654  000034          MOV     #POWDWN,@DVEC   ;RESET THE DOWN VECTOR
005010  000113                          JMP     (R3)            ;JUMP INDIRECT TO R3

005012  104401                  ILLUP:  TYPE                    ;POWER UP BEFORE POWER DOWN COMPLETE
005014  010055                          UPF
005016  000000                          HALT
005020  000167  173154                  JMP     200
005024  104401                  ILLDWN: TYPE                    ;POWER DOWN BEFORE UP COMPLETE
005026  010125                          DOWN
005030  000000                          HALT
005032  000167  173142                  JMP     200

005036  000000                  ERRAD:  0                       ;RETURN ADDRESS FROM POWER FAIL

005040  000024  000026          UVEC:   24,26                   ;UP ADDRESS PAIR
005044  000024  000026          DVEC:   24,26                   ;DOWN ADDRESS PAIR
005050  000000                  SAV6:   0                       ;SOME PLACE TO PUT THE SP
005052  000000                  FLAG:   0                       ;1 INSTRUCTION DOWN FLAG
005054  000000                  LIMIT:  0                       ;TOP OF MEMORY
005056  052525                  DATA:   52525                   ;WHAT IS TO BE WRITTEN INTO MEMORY
```

```
                                        .SBTTL  SETUP MEMORY MANAGMENT REGISTERS

005060  012737  000000  172340  MAP:    MOV     #0,@#KIPAR0     ;SETUP PAR0 FOR 1ST 4K
005066  012737  077406  172300          MOV     #77406,@#KIPDR0 ;4K, R/W, EXPAND UP
005074  012737  000000  172352          MOV     #0,@#KIPAR5     ;SET UP PAR5 FOR 1ST 4K
005102  012737  077406  172312          MOV     #77406,@#KIPDR5 ;4K, R/W, ED=UP
005110  012737  000200  172354          MOV     #200,@#KIPAR6   ;SET UP PAR6 FOR 2ND 4K
005116  012737  000000  172314          MOV     #0,@#KIPDR6     ;ABORT ALL REFERENCES
005124  012737  177600  172356          MOV     #177600,@#KIPAR7  ;SET UP PAR7 FOR I/O PAGE
005132  012737  077406  172316          MOV     #77406,@#KIPDR7 ;4K, R/W, ED=UP
005140  012737  000020  172516          MOV     #BIT04,@#MMR3   ;SET UP FOR 22-BIT MAPPING
005146  012737  005156  000250          MOV     #MMERR,@#MMVEC  ;SET UP MEMORY MANAGEMENT VECTOR
005154  000207                          RTS     PC              ;RETURN FROM CALL
005156  000000                  MMERR:  HALT                    ;MEMORY MANAGEMENT ERROR
005160  000776                          BR      MMERR
                                        .SBTTL ROUTINE TO SIZE MEMORY
                                        .SBTTL  ROUTINE TO SIZE MEMORY

                                        ;;**********************************************************************
                                        ;*CALL:
                                        ;*      JSR     PC,$SIZE
                                        ;*      RETURN
                                        ;*$LSTAD WILL CONTAIN:
                                        ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
                                        ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                        ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                        ;*$KT11 IS THE MEMORY MANAGEMENT KEY
                                        ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                        ;*      MUST BE SETUP BEFORE THE CALL
                                        ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
                                        ;*      DETERMINED BY ROUTINE

005162  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
005164  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
005166  010246                          MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
005170  010346                          MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
005172  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
005176  013746  000006                  MOV     @#ERRVEC+2,-(SP)
005202  010600                          MOV     SP,R0           ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
005204  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
005206  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
005212  012701  003776                  MOV     #3776,R1        ;;SETUP ADDRESS
005216  105727                          TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
005220  000200                  $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
005222  100062                          BPL     $CORE           ;;BR IF NO
005224  012737  005362  000004          MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
005232  005737  177572                  TST     @#SR0           ;;KT11 ARE YOU THERE?
005236  052767  100000  177754          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
005244  005046                          CLR     -(SP)           ;;INITIALIZE FOR 'PAR' LOADING
005246  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST 'PAR'
005252  012703  000010                  MOV     #^D8,R3         ;;LOAD EIGHT 'PAR.'S AND EIGHT 'PDR.'S
005256  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
005264  011622                          MOV     (SP),(R2)+      ;;LOAD 'PAR'
005266  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT 'PAR'
005272  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
005274  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
```

```
005300  005042                          CLR     -(R2)                   ;;SETUP KIPAR6 FOR TESTING
005302  012737  005320  000004          MOV     #2$,@#ERRVEC            ;;CATCH TIMEOUT IF NO SR3
005310  012737  000020  172516          MOV     #20,@#SR3               ;;ENABLE 22 BIT MODE
005316  000401                          BR      3$                      ;;THIS PDP-11 HAS A SR3 REGISTER
005320  022626                  2$:     CMP     (SP)+,(SP)+             ;;CLEAN OFF THE STACK--NO SR3
005322  005237  177572          3$:     INC     @#SR0                   ;;TURN ON MEMORY MANAGEMENT
005326  012737  005352  000004          MOV     #$KTOUT,@#ERRVEC        ;;SET FOR TIME OUT
005334  005737  143776          4$:     TST     @#143776                ;;TRAP ON NON-EX-MEM
005340  062712  000040                  ADD     #40,(R2)                ;;MAKE A 1K STEP
005344  023712  172356                  CMP     @#KIPAR7,(R2)           ;;LAST ONE?
005350  101371                          BHI     4$                      ;;NO--TRY IT
005352  011202                  $KTOUT: MOV     (R2),R2                 ;;GET LAST BANK+1
005354  005037  177572                  CLR     @#SR0                   ;;TURN OFF MEMORY MANAGEMENT
005360  000421                          BR      $SIZEX
005362  042767  100000  177630  $KTNEX: BIC     #100000,$KT11           ;;KT11 NON-EXISTENT
005370  012737  005420  000004  $CORE:  MOV     #$CROUT,@#ERRVEC        ;;SET FOR TIMEOUT
005376  005002                          CLR     R2                      ;;SET UP BANK
005400  062701  004000          1$:     ADD     #4000,R1                ;;INCREMENT BY 1K
005404  062702  000040                  ADD     #40,R2                  ;;1K STEP
005410  005711                          TST     (R1)                    ;;TRAP ON TIME OUT
005412  022701  177776                  CMP     #177776,R1              ;;LAST ONE
005416  001370                          BNE     1$                      ;;NO--TRY AGAIN
005420  162701  004000  $CROUT: SUB     #4000,R1
005424  162702  000040  $SIZEX: SUB     #40,R2                  ;;DROP BACK
005430  010006                          MOV     R0,SP                   ;;RESTORE THE STACK
005432  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;RESTORE ERROR VECTOR
005436  012637  000004                  MOV     (SP)+,@#ERRVEC
005442  010167  000016                  MOV     R1,$LSTAD               ;;LAST ADDRESS
005446  010267  000014                  MOV     R2,$LSTBK               ;;LAST BANK
005452  012603                          MOV     (SP)+,R3                ;;RESTORE R3
005454  012602                          MOV     (SP)+,R2                ;;RESTORE R2
005456  012601                          MOV     (SP)+,R1                ;;RESTORE R1
005460  012600                          MOV     (SP)+,R0                ;;RESTORE R0
005462  000207                          RTS     PC
005464  000000          $LSTAD: .WORD   0                       ;;CONTAINS THE LAST ADDRESS
005466  000000          $LSTBK: .WORD   0                       ;;CONTAINS THE LAST BANK
811
812
```

```
814                                     .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                                      ;;1=TYPE LEADING ZEROS
                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT

005470  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
005474  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
005502  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
005506  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
005512  000406                          BR      $TYPON
005514  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
005522  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
005530  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
005536  010346                          MOV     R3,-(SP)        ;;SAVE R3
005540  010446                          MOV     R4,-(SP)        ;;SAVE R4
005542  010546                          MOV     R5,-(SP)        ;;SAVE R5
005544  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
005550  005404                          NEG     R4
005552  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
005556  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
005562  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
005566  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
005572  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
005574  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO ''C''
005576  000404                          BR      3$              ;;GO DO MSB
005600  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
005602  006105                          ROL     R5
005604  006105                          ROL     R5
005606  010503                          MOV     R5,R3
005610  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
005612  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
005616  100016                          BPL     7$              ;;BR IF NO
005620  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
005624  001002                          BNE     4$              ;;TEST FOR 0
005626  005704                          TST     R4              ;;SUPPRESS THIS 0?
005630  001403                          BEQ     5$              ;;BR IF YES
005632  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
005634  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
005640  052703  000040      5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
005644  110367  000040              MOVB    R3,8$           ;;SAVE FOR TYPING
005650  104401  005710              TYPE    ,8$             ;;GO TYPE THIS DIGIT
005654  105367  000032      7$:     DECB    $OCNT           ;;COUNT BY 1
005660  003347                      BGT     2$              ;;BR IF MORE TO DO
005662  002402                      BLT     6$              ;;BR IF DONE
005664  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
005666  000744                      BR      2$              ;;GO DO THE LAST DIGIT
005670  012605      6$:             MOV     (SP)+,R5        ;;RESTORE R5
005672  012604                      MOV     (SP)+,R4        ;;RESTORE R4
005674  012603                      MOV     (SP)+,R3        ;;RESTORE R3
005676  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
005704  012616                      MOV     (SP)+,(SP)
005706  000002                      RTI                     ;;RETURN
005710     000      8$:             .BYTE   0               ;;STORAGE FOR ASCII DIGIT
005711     000                      .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
005712     000      $OCNT:          .BYTE   0               ;;OCTAL DIGIT COUNTER
005713     000      $OFILL:         .BYTE   0               ;;ZERO FILL SWITCH
005714  000000      $OMODE:         .WORD   0               ;;NUMBER OF DIGITS TO TYPE
  815               .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                    ;;**********************************************************************
                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                    ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                    ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                    ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                    ;*REPLACED WITH SPACES.
                    ;*CALL:
                    ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                    ;*      TYPDS                   ;;GO TO THE ROUTINE

005716              $TYPDS:
005716  010046              MOV     R0,-(SP)        ;;PUSH R0 ON STACK
005720  010146              MOV     R1,-(SP)        ;;PUSH R1 ON STACK
005722  010246              MOV     R2,-(SP)        ;;PUSH R2 ON STACK
005724  010346              MOV     R3,-(SP)        ;;PUSH R3 ON STACK
005726  010546              MOV     R5,-(SP)        ;;PUSH R5 ON STACK
005730  012746  020200      MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
005734  016605  000020      MOV     20(SP),R5       ;;GET THE INPUT NUMBER
005740  100004              BPL     1$              ;;BR IF INPUT IS POS.
005742  005405              NEG     R5              ;;MAKE THE BINARY NUMBER POS.
005744  112766  000055  000001  MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
005752  005000      1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
005754  012703  006132      MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
005760  112723  000040      MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
005764  005002      2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
005766  016001  006122      MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
005772  160105      3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
005774  002402              BLT     4$              ;;BR IF DONE
005776  005202              INC     R2              ;;INCREASE THE BCD DIGIT BY 1
006000  000774              BR      3$
006002  060105      4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
006004  005702              TST     R2              ;;CHECK IF BCD DIGIT=0
006006  001002              BNE     5$              ;;FALL THROUGH IF 0
006010  105716              TSTB    (SP)            ;;STILL DOING LEADING 0'S?
006012  100407              BMI     7$              ;;BR IF YES
```

```
006014  106316                  5$:     ASLB    (SP)            ;;MSD?
006016  103003                          BCC     6$              ;;BR IF NO
006020  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
006026  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
006032  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
006036  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
006040  005720                          TST     (R0)+           ;;JUST INCREMENTING
006042  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
006046  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
006050  003002                          BGT     8$              ;;GO TO EXIT
006052  010502                          MOV     R5,R2           ;;GET THE LSD
006054  000764                          BR      6$              ;;GO CHANGE TO ASCII
006056  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
006060  100003                          BPL     9$              ;;BR IF NO
006062  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
006070  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
006072  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
006074  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
006076  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
006100  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
006102  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
006104  104401  006132                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
006110  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
006116  012616                          MOV     (SP)+,(SP)
006120  000002                          RTI                     ;;RETURN TO USER
006122  023420                  $DTBL:  10000.
006124  001750                          1000.
006126  000144                          100.
006130  000012                          10.
006132                          $DBLK:  .BLKW   4
  816                           .SBTTL  TYPE ROUTINE

                                ;;******************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

006142  105767  000335          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
006146  100002                          BPL     1$              ;;BR IF YES
006150  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
006152  000430                          BR      3$              ;;LEAVE
006154  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
006156  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
006162  122767  000001  172744          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
006170  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
006172  132767  000100  172735          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
006200  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
```

```
006202  010067  000004                 MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
006206  004767  172744                 JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
006212  000000                 61$:    .WORD   0               ;;MESSAGE ADDRESS
006214  132767  000040  172713 62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
006222  001003                         BNE     60$             ;;YES,SKIP TYPE OUT
006224  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
006226  001005                         BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
006230  005726                         TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
006232  012600                 60$:    MOV     (SP)+,R0        ;;RESTORE R0
006234  062716  000002         3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
006240  000002                         RTI                     ;;RETURN
006242  122716  000011         4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
006246  001430                         BEQ     8$
006250  122716  000200                 CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
006254  001006                         BNE     5$
006256  005726                         TST     (SP)+           ;;POP  <CR><LF> EQUIV
006260  104401                         TYPE                    ;;TYPE A CR AND LF
006262  006505                         $CRLF
006264  105067  000200                 CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
006270  000755                         BR      2$              ;;GET NEXT CHARACTER
006272  004767  000056         5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
006276  126726  000200         6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
006302  001350                         BNE     2$              ;;IF NO GO GET NEXT CHAR.
006304  016746  000170                 MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                               ;;AND THE NULL CHAR.
006310  105366  000001         7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
006314  002770                         BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
006316  004767  000032                 JSR     PC,$TYPEC       ;;GO TYPE A NULL
006322  105367  000142                 DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
006326  000770                         BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

006330  112716  000040         8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
006334  004767  000014         9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
006340  132767  000007  000122         BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
006346  001372                         BNE     9$              ;;TAB STOP
006350  005726                         TST     (SP)+           ;;POP SPACE OFF STACK
006352  000724                         BR      2$              ;;GET NEXT CHARACTER
006354  105777  000114         $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
006360  100375                         BPL     $TYPEC
006362  116677  000002  000106         MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
006370  105777  000114                 TSTB    @$TKS           ;;SEE IF KEYBOARD IS TALKING.
006374  100021                         BPL     2$              ;;BRANCH IF IT ISN'T.
006376  017746  000110                 MOV     @$TKB,-(SP)     ;;PUSH CHARACTER ONTO STACK.
006402  042716  177600                 BIC     #177600,(SP)    ;;BIT CLEAR TOP BYTE AND PARITY BIT.
006406  022726  000023                 CMP     #23,(SP)+       ;;SEE IF THIS IS A ^S.
006412  001012                         BNE     2$              ;;BRANCH TO CONTINUE IF IT ISN'T.
006414  105777  000070         3$:     TSTB    @$TKS           ;;WAIT FOR ANOTHER INPUT.
006420  100375                         BPL     3$              ;;BRANCH BACK IF NOT READY.
006422  017746  000064                 MOV     @$TKB,-(SP)     ;;PUSH NEXT CHARACTER ON STACK.
006426  042716  177600                 BIC     #177600,(SP)    ;;BIT CLEAR TOP BYTE AND PARITY BIT.
006432  022726  000021                 CMP     #21,(SP)+       ;;SEE IF THIS IS A ^Q.
006436  001366                         BNE     3$              ;;BRANCH BACK FOR MORE WAIT IF NOT.
006440  122766  000015  000002 2$:     CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
006446  001003                         BNE     1$              ;;BRANCH IF NO
006450  105067  000014                 CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
```

```
006454  000406                          BR      $TYPEX          ;;EXIT
006456  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
006464  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
006466  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
006470  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
006472  000207                  $TYPEX: RTS     PC

006474  177564                  $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
006476  177566                  $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
006500     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
006501     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
006502     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
006503     000                  $TPFLG: .BYTE   0               ;;''TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
006504     077                  $QUES:  .ASCII  ''?''           ;;QUESTION MARK
006505     015                  $CRLF:  .ASCII  <15>            ;;CARRAIGE RETURN
006506     012      000         $LF:    .ASCIZ  <12>            ;;LINEFEED
817 006510  177560              $TKS:   .WORD   177560
818 006512  177562              $TKB:   .WORD   177562
819
820                             .SBTTL  TRAP DECODER

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

006514  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
006516  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
006522  005740                          TST     -(R0)           ;;BACKUP BY 2
006524  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
006526  006300                          ASL     R0              ;;POSITION FOR INDEXING
006530  016000  006550                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
006534  000200                          RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO

006536  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
006540  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
006546  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE ''TRAP'' INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
006550  006536                  $TRPAD: .WORD   $TRAP2
006552  006142                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
006554  005514                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
006556  005470                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
006560  005530                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
006562  005716                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
```

```
                                ;-----------------------------------------
                                ;       ERROR PRINT ROUTINE
                                ;-----------------------------------------
                        104000          ERROR=104000

006564  000000                  ERRPC:  .WORD   0

006566  010067  000026  $ERROR:         MOV R0,SAVR0            ;SAVE R0 THRU R5
006572  010167  000024                  MOV R1,SAVR1
006576  010267  000022                  MOV R2,SAVR2
006602  010367  000020                  MOV R3,SAVR3
006606  010467  000016                  MOV R4,SAVR4
006612  010567  000014                  MOV R5,SAVR5
006616  000406                        BR TITL
006620  000000                  SAVR0:  .WORD
006622  000000                  SAVR1:  .WORD
006624  000000                  SAVR2:  .WORD
006626  000000                  SAVR3:  .WORD
006630  000000                  SAVR4:  .WORD
006632  000000                  SAVR5:  .WORD

006634  011601  TITL:                   MOV (SP),R1             ;R1 CONTAINS ADDRESS FLLOWING ERRPC ADDRESS
006636  162701  000002                SUB #2,R1
006642  010167  177716                MOV R1,ERRPC            ;GET ADDRESS OF ERROR MESSAGE:ERRPC
006646  104401  006654          TYPE    ,65$            ;;TYPE ASCIZ STRING
006652  000411                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <CRLF><CRLF>/TESTNO    ERRPC/
006676                  64$:
006676  005721          3$:     TST (R1)+               ;R1 POINTS TO NEXT ARGUEMENT

006700  012702  007652          MOV #PRTABL,R2          ;ADDRESS OF START OF PRINT TABLE LIST
006704  012703  007126          MOV #PRTITL,R3          ;ADDRESS OF STERT OF TITLES
006710  005711                  TST (R1)                ;END OF ARGUEMENTS?
006712  001405                  BEQ DAT                 ;YES,GO PRINT DATA
006714  005723          2$:     TST (R3)+               ;INDEX THRU TITLES
006716  021122                  CMP (R1),(R2)+          ;SEARCH PRINT TABLE LIST FOR TITLE
006720  001375                  BNE 2$                  ;NO; CHECK NEXT LOCATION IN LIST

006722  004753                  JSR PC,@-(R3)           ;FOUND IT; GO PRINT TITLE
                                                        ;CODE ERROR ROUTINE
006724  000764                  BR 3$

006726                  DAT:
006726  104401  006734          TYPE    ,65$            ;;TYPE ASCIZ STRING
006732  000401                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <CRLF>
```

```
                006736
    872 006736  016746  172156          64$:    MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
        006742  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
        006744     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
        006745     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
    873 006746  104401  006754                  TYPE    ,67$            ;;TYPE ASCIZ STRING
        006752  000402                          BR      66$             ;;GET OVER THE ASCIZ
                                            ;;67$:  .ASCIZ  /  /
                006760                      66$:
    874 006760  016746  177600                  MOV     ERRPC,-(SP)     ;;SAVE ERRPC FOR TYPEOUT
        006764  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    875 006766  104401  004234                  TYPE ,$ENULL
    876
    877 006772  011601                          MOV (SP),R1             ;R1 CONTAINS ADDRESS FOLLOWING ERRORPC ADDRESS
    878 006774  162701  000002                  SUB #2,R1
    879 007000  005721                  3$:     TST (R1)+               ;R1 POINTS TO NEXT ARGUEMENT
    880 007002  012702  007652                  MOV #PRTABL,R2          ;ADDRESS OF START OF PRINT TABLE LIST
    881 007006  012703  007506                  MOV #PRDATA,R3          ;ADDRESS OF STERT OF PRINT DATA
    882 007012  005711                          TST (R1)                ;END OF ARGUEMENTS?
    883 007014  001421                          BEQ FIN                 ;YES
    884 007016  005723                  2$:     TST (R3)+               ;INDEX THRU DATA PRINTS
    885 007020  021122                          CMP (R1),(R2)+          ;SEARCH PRINT TABLE LIST FOR TITLE
    886 007022  001375                          BNE 2$                  ;NO; CHECK NEXT LOCATION IN LIST
    887 007024  104401  007032                  TYPE    ,69$            ;;TYPE ASCIZ STRING
        007030  000402                          BR      68$             ;;GET OVER THE ASCIZ
                                            ;;69$:  .ASCIZ  /   /
                007036                      68$:
    888 007036  004753                          JSR PC,@-(R3)           ;
    889 007040  104401  004234                  TYPE ,$ENULL
    890 007044  104401  007052                  TYPE    ,71$            ;;TYPE ASCIZ STRING
        007050  000402                          BR      70$             ;;GET OVER THE ASCIZ
                                            ;;71$:  .ASCIZ  /   /
                007056                      70$:
    891 007056  000750                          BR 3$
    892
    893 007060  005721                  FIN:    TST (R1)+               ;R1 NOW CONTAINS RETURN ADDRESS
    894 007062  010116                          MOV R1,(SP)             ;SETUP RETURN ADDRESS IN STACK
    895 007064  104401  007072                  TYPE    ,65$            ;;TYPE ASCIZ STRING
        007070  000401                          BR      64$             ;;GET OVER THE ASCIZ
                                            ;;65$:  .ASCIZ  <CRLF>
                007074                      64$:
    896
    897 007074  016700  177520                  MOV SAVR0,R0            ;RESTORE REGISTERS
    898 007100  016701  177516                  MOV SAVR1,R1
    899 007104  016702  177514                  MOV SAVR2,R2
    900 007110  016703  177512                  MOV SAVR3,R3
    901 007114  016704  177510                  MOV SAVR4,R4
    902 007120  016705  177506                  MOV SAVR5,R5
    903
    904 007124  000002                          RTI                     ;RETURN
    905 007126  007152                  PRTITL: 1$
    906 007130  007200                          2$
    907 007132  007226                          3$
    908 007134  007254                          4$
    909 007136  007302                          5$
    910 007140  007330                          6$
    911 007142  007356                          7$
```

```
                                                                E 4
    912 007144  007404                          10$
    913 007146  007432                          11$
    914 007150  007460                          12$
    915 007152                                  1$:
        007152  104401  007160                  TYPE    ,65$            ;;TYPE ASCIZ STRING
        007156  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  /   EXPDAT   /
        007176                          64$:
    916 007176  000207                          RTS PC
    917 007200                                  2$:
        007200  104401  007206                  TYPE    ,67$            ;;TYPE ASCIZ STRING
        007204  000407                          BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  /   RECDAT   /
        007224                          66$:
    918 007224  000207                          RTS PC
    919 007226                                  3$:
        007226  104401  007234                  TYPE    ,69$            ;;TYPE ASCIZ STRING
        007232  000407                          BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  /   EXP.ADD   /
        007252                          68$:
    920 007252  000207                          RTS PC
    921 007254                                  4$:
        007254  104401  007262                  TYPE    ,71$            ;;TYPE ASCIZ STRING
        007260  000407                          BR      70$             ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  /   REC.ADD   /
        007300                          70$:
    922 007300  000207                          RTS PC
    923 007302                                  5$:
        007302  104401  007310                  TYPE    ,73$            ;;TYPE ASCIZ STRING
        007306  000407                          BR      72$             ;;GET OVER THE ASCIZ
                                        ;;73$:  .ASCIZ  /   EXP.SP1   /
        007326                          72$:
    924 007326  000207                          RTS PC
    925 007330                                  6$:
        007330  104401  007336                  TYPE    ,75$            ;;TYPE ASCIZ STRING
        007334  000407                          BR      74$             ;;GET OVER THE ASCIZ
                                        ;;75$:  .ASCIZ  /   REC.SP1   /
        007354                          74$:
    926 007354  000207                          RTS PC
    927 007356                                  7$:
        007356  104401  007364                  TYPE    ,77$            ;;TYPE ASCIZ STRING
        007362  000407                          BR      76$             ;;GET OVER THE ASCIZ
                                        ;;77$:  .ASCIZ  /   EXP.PSW   /
        007402                          76$:
    928 007402  000207                          RTS PC
```

```
    930 007404                                  10$:
        007404  104401  007412                  TYPE    ,79$            ;;TYPE ASCIZ STRING
        007410  000407                          BR      78$             ;;GET OVER THE ASCIZ
                                        ;;79$:  .ASCIZ  /   REC.PSW   /
        007430                          78$:
    931 007430  000207                          RTS PC
    932 007432                                  11$:
        007432  104401  007440                  TYPE    ,81$            ;;TYPE ASCIZ STRING
        007436  000407                          BR      80$             ;;GET OVER THE ASCIZ
                                        ;;81$:  .ASCIZ  /   VIR.ADD   /
        007456                          80$:
```

```
934 007456  000207                    RTS PC
```

```
 936 007460                                    12$:
     007460  104401  007466                    TYPE    ,83$            ;;TYPE ASCIZ STRING
     007464  000407                            BR      82$             ;;GET OVER THE ASCIZ
                                        ;;83$:  .ASCIZ  /   PAR.OFF   /
     007504                             82$:
 937 007504  000207                            RTS PC
 938                                           .EVEN
 939
 940
 941
 942 007506  007532                 PRDATA:    1$
 943 007510  007542                            2$
 944 007512  007552                            3$
 945 007514  007562                            4$
 946 007516  007572                            5$
 947 007520  007602                            6$
 948 007522  007612                            7$
 949 007524  007622                            10$
 950 007526  007632                            11$
 951 007530  007642                            12$
 952 007532                                    1$:
     007532  016746  000140                    MOV     EXPDAT,-(SP)    ;;SAVE EXPDAT FOR TYPEOUT
     007536  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 953 007540  000207                            RTS PC
 954 007542                                    2$:
     007542  016746  000132                    MOV     RECDAT,-(SP)    ;;SAVE RECDAT FOR TYPEOUT
     007546  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 955 007550  000207                            RTS PC
 956 007552                                    3$:
     007552  016746  000124                    MOV     EXP.ADD,-(SP)   ;;SAVE EXP.ADD FOR TYPEOUT
     007556  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 957 007560  000207                            RTS PC
 958 007562                                    4$:
     007562  016746  000116                    MOV     REC.ADD,-(SP)   ;;SAVE REC.ADD FOR TYPEOUT
     007566  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 959 007570  000207                            RTS PC
 960 007572                                    5$:
     007572  016746  000110                    MOV     EXP.SP1,-(SP)   ;;SAVE EXP.SP1 FOR TYPEOUT
     007576  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 961 007600  000207                            RTS PC
 962 007602                                    6$:
     007602  016746  000102                    MOV     REC.SP1,-(SP)   ;;SAVE REC.SP1 FOR TYPEOUT
     007606  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 963 007610  000207                            RTS PC
 964 007612                                    7$:
     007612  016746  000074                    MOV     EXP.PSW,-(SP)   ;;SAVE EXP.PSW FOR TYPEOUT
     007616  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 965 007620  000207                            RTS PC
 966 007622                                    10$:
     007622  016746  000066                    MOV     REC.PSW,-(SP)   ;;SAVE REC.PSW FOR TYPEOUT
     007626  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 967 007630  000207                            RTS PC
 968 007632                                    11$:
     007632  016746  000060                    MOV     VIR.ADD,-(SP)   ;;SAVE VIR.ADD FOR TYPEOUT
     007636  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 969 007640  000207                            RTS PC
 970 007642                                    12$:
```

```
        007642  016746  000052                  MOV     PAR.OFF,-(SP)   ;;SAVE PAR.OFF FOR TYPEOUT
        007646  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
    971 007650  000207                                  RTS PC
    972
    973
    974 007652                          PRTABL:
    975 007652  007676                          EXPDAT
    976 007654  007700                          RECDAT
    977 007656  007702                          EXP.ADD
    978 007660  007704                          REC.ADD
    979 007662  007706                          EXP.SP1
    980 007664  007710                          REC.SP1
    981 007666  007712                          EXP.PSW
    982 007670  007714                          REC.PSW
    983 007672  007716                          VIR.ADD
    984 007674  007720                          PAR.OFF
    985
```

```
                                                EXPDAT: .WORD   0
                                                RECDAT: .WORD   0
                                                EXP.ADD: .WORD  0
                                                REC.ADD: .WORD  0
                                                EXP.SP1: .WORD  0
                                                REC.SP1: .WORD  0
                                                EXP.PSW: .WORD  0
                                                REC.PSW: .WORD  0
                                                VIR.ADD: .WORD  0
                                                PAR.OFF: .WORD  0
 998 007722   015   012   103                          PNAME: .ASCIZ <15><12>+CKKACA0 11/44 POWER FAIL+
     007725   113   113   101
     007730   103   101   060
     007733   040   061   061
     007736   057   064   064
     007741   040   120   117
     007744   127   105   122
     007747   040   106   101
     007752   111   114   000
 999 007755   015   012   103                          TITLE: .ASCIZ <15><12>+CKKACA0 11/44 POWER FAIL+<15><12>
     007760   113   113   101
     007763   103   101   060
     007766   040   061   061
     007771   057   064   064
     007774   040   120   117
     007777   127   105   122
     010002   040   106   101
     010005   111   114   015
     010010   012   000
1000 010012   015   012   102                          BOOT:  .ASCIZ <15><12>+BOOT ENABLE SWITCH MUST BE OFF+<15><12>
     010015   117   117   124
     010020   040   105   116
     010023   101   102   114
     010026   105   040   123
     010031   127   111   124
     010034   103   110   040
     010037   115   125   123
     010042   124   040   102
     010045   105   040   117
     010050   106   106   015
     010053   012   000
1001 010055   015   012   120                          UPF: .ASCIZ <15><12>+POWER UP BEFORE POWER DOWN COMPLETE+<15><12>
     010060   117   127   105
     010063   122   040   125
     010066   120   040   102
     010071   105   106   117
     010074   122   105   040
     010077   120   117   127
     010102   105   122   040
     010105   104   117   127
     010110   116   040   103
     010113   117   115   120
     010116   114   105   124
     010121   105   015   012
     010124   000
1002
1003 010125   015   012   120                          DOWN: .ASCIZ <15><12>+POWER DOWN BEFORE UP COMPLETE+<15><12>
```

```
        010130      117     127     105
        010133      122     040     104
        010136      117     127     116
        010141      040     102     105
        010144      106     117     122
        010147      105     040     125
        010152      120     040     103
        010155      117     115     120
        010160      114     105     124
        010163      105     015     012
        010166      000
  1004
  1005
  1006
  1007
  1008
  1009                                      .EVEN
  1010 010170  000000                       END:    0
  1011         000001                       .END
```

```
ABASE = 000000
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APF     002134
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000
BEGIN   001424
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BOOT    010012
BPTVEC= 000014
CACHVE= 000114
CHECK   004360
CONTRL= 177746
CPUERR= 177766
CR    = 000015
CRLF  = 000200
DAT     006726
DATA    005056
DDISP = 177570
DISPLA= 177570
DISPRE  000174
DOWN    010125
DSWR  = 177570
DVEC    005044
EMTVEC= 000030
END     010170
ERRAD   005036
ERROR = 104000
ERRPC   006564
ERRVEC= 000004
EXPDAT  007676
EXP.AD  007702
EXP.PS  007712
EXP.SP  007706
FIN     007060
FLAG    005052
FT      002264
HIADRS= 177742
HITMIS= 177752
HT    = 000011
ILLDWN  005024
ILLUP   005012
IOTVEC= 000020
KDPAR0= 172360
KDPAR1= 172362
KDPAR2= 172364
KDPAR3= 172366
KDPAR4= 172370
KDPAR5= 172372
KDPAR6= 172374
KDPAR7= 172376
KDPDR0= 172320
KDPDR1= 172322
KDPDR2= 172324
KDPDR3= 172326
KDPDR4= 172330
KDPDR5= 172332
KDPDR6= 172334
KDPDR7= 172336
KERSTK= 001100
KIPAR0= 172340
KIPAR1= 172342
KIPAR2= 172344
KIPAR3= 172346
KIPAR4= 172350
KIPAR5= 172352
KIPAR6= 172354
KIPAR7= 172356
KIPDR0= 172300
KIPDR1= 172302
KIPDR2= 172304
KIPDR3= 172306
KIPDR4= 172310
KIPDR5= 172312
KIPDR6= 172314
KIPDR7= 172316
KSP   =%000006
LF    = 000012
LIMIT   005054
LKS   = 177546
LKVEC = 000100
LOAD    004254
LOADRS= 177740
LRTI    001422
MAINT = 177750
MAP     005060
MAPH0 = 170202
MAPH00= 170202
MAPH01= 170206
MAPH02= 170212
MAPH03= 170216
MAPH04= 170222
MAPH05= 170226
MAPH06= 170232
MAPH07= 170236
MAPH1 = 170206
MAPH10= 170242
MAPH11= 170246
MAPH12= 170252
MAPH13= 170256
MAPH14= 170262
MAPH15= 170266
MAPH16= 170272
MAPH17= 170276
MAPH2 = 170212
MAPH20= 170302
MAPH21= 170306
MAPH22= 170312
MAPH23= 170316
MAPH24= 170320
MAPH25= 170326
MAPH26= 170332
MAPH27= 170336
MAPH3 = 170216
MAPH30= 170342
MAPH31= 170346
MAPH32= 170352
MAPH33= 170356
MAPH34= 170362
MAPH35= 170366
MAPH36= 170372
MAPH37= 170376
MAPH4 = 170222
MAPH5 = 170226
MAPH6 = 170232
MAPH7 = 170236
MAPL0 = 170200
MAPL00= 170200
MAPL01= 170204
MAPL02= 170210
MAPL03= 170214
MAPL04= 170220
MAPL05= 170224
MAPL06= 170230
MAPL07= 170234
MAPL1 = 170204
MAPL10= 170240
MAPL11= 170244
MAPL12= 170250
MAPL13= 170254
MAPL14= 170260
MAPL15= 170264
MAPL16= 170270
MAPL17= 170274
MAPL2 = 170210
MAPL20= 170300
MAPL21= 170304
MAPL22= 170310
MAPL23= 170314
MAPL24= 170320
MAPL25= 170324
MAPL26= 170330
MAPL27= 170334
MAPL3 = 170214
MAPL30= 170340
MAPL31= 170344
MAPL32= 170350
MAPL33= 170354
MAPL34= 170360
MAPL35= 170364
MAPL36= 170370
MAPL37= 170374
MAPL4 = 170220
MAPL5 = 170224
MAPL6 = 170230
MAPL7 = 170234
MEMERR= 177744
MMERR   005156
MMR0  = 177572
MMR1  = 177574
MMR2  = 177576
MMR3  = 172516
MMVEC = 000250
PARERR  004554
PARFLG  004556
PAR.OF  007720
PFAIL   001416
PIRQ  = 177772
PIRQVE= 000240
PNAME   007722
POWDWN  004654
POWUP   004750
PRDATA  007506
PRETST  002274
PRTABL  007652
PRTITL  007126
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PWRVEC= 000024
RECDAT  007700
REC.AD  007704
REC.PS  007714
REC.SP  007710
RESVEC= 000010
R10   =%000000
R11   =%000001
R12   =%000002
R13   =%000003
R14   =%000004
R15   =%000005
R6    =%000006
R7    =%000007
SAVR0   006620
SAVR1   006622
SAVR2   006624
```

```
SAVR3    006626          STACK1= 000500          TST5    003370          XYZ     001420          $MSGAD  001130
SAVR4    006630          STKLMT= 177774          TST6    003500          $APTHD  001100          $MSGLG  001132
SAVR5    006632          SUPSTK= 000700          TST7    003620          $ATYC   001174          $MSGTY  001114
SAV6     005050          SWR    = 177570          TYPDS = 104405          $ATY1   001150          $MTYP1  001145
SCOPE  = 000004          SWREG   000176          TYPE  = 104401          $ATY3   001156          $NULL   006500
SDPAR0= 172260           SW0    = 000001          TYPOC = 104402          $ATY4   001166          $NWTST= 000001
SDPAR1= 172262           SW00   = 000001          TYPON = 104404          $CHARC  006470          $OCNT   005712
SDPAR2= 172264           SW01   = 000002          TYPOS = 104403          $CORE   005370          $OMODE  005714
SDPAR3= 172266           SW02   = 000004          UDPAR0= 177660          $CPUOP  001142          $PASS   001122
SDPAR4= 172270           SW03   = 000010          UDPAR1= 177662          $CRLF   006505          $PASTM  001106
SDPAR5= 172272           SW04   = 000020          UDPAR2= 177664          $CROUT  005420          $QUES   006504
SDPAR6= 172274           SW05   = 000040          UDPAR3= 177666          $DBLK   006132          $RTNAD  004232
SDPAR7= 172276           SW06   = 000100          UDPAR4= 177670          $DEVCT  001124          $SAVPC= 000204
SDPDR0= 172220           SW07   = 000200          UDPAR5= 177672          $DOAGN  004230          $SETUP= 000024
SDPDR1= 172222           SW08   = 000400          UDPAR6= 177674          $DTBL   006122          $SIZE   005162
SDPDR2= 172224           SW09   = 001000          UDPAR7= 177676          $ENDAD  004220          $SIZEX  005424
SDPDR3= 172226           SW1    = 000002          UDPDR0= 177620          $ENDCT  004166          $STUP = 177777
SDPDR4= 172230           SW10   = 002000          UDPDR1= 177622          $ENDMG  004237          $SVPC = 000204
SDPDR5= 172232           SW11   = 004000          UDPDR2= 177624          $ENULL  004234          $SWR  = 040000
SDPDR6= 172234           SW12   = 010000          UDPDR3= 177626          $ENV    001134          $SWREG  001136
SDPDR7= 172236           SW13   = 020000          UDPDR4= 177630          $ENVM   001135          $TESTN  001120
SIPAR0= 172240           SW14   = 040000          UDPDR5= 177632          $EOP    004142          $TKB    006512
SIPAR1= 172242           SW15   = 100000          UDPDR6= 177634          $EOPCT  004160          $TKS    006510
SIPAR2= 172244           SW2    = 000004          UDPDR7= 177636          $ERROR  006566          $TN   = 000012
SIPAR3= 172246           SW3    = 000010          UIPAR0= 177640          $ETABL  001134          $TPB    006476
SIPAR4= 172250           SW4    = 000020          UIPAR1= 177642          $ETEND  001150          $TPFLG  006503
SIPAR5= 172252           SW5    = 000040          UIPAR2= 177644          $FATAL  001116          $TPS    006474
SIPAR6= 172254           SW6    = 000100          UIPAR3= 177646          $FFLG   001414          $TRAP   006514
SIPAR7= 172256           SW7    = 000200          UIPAR4= 177650          $FILLC  006502          $TRAP2  006536
SIPDR0= 172200           SW8    = 000400          UIPAR5= 177652          $FILLS  006501          $TRP  = 000006
SIPDR1= 172202           SW9    = 001000          UIPAR6= 177654          $GET42  004210          $TRPAD  006550
SIPDR2= 172204           SYSTID= 177764           UIPAR7= 177656          $HD   = 000000          $TSTM   001104
SIPDR3= 172206           TBITVE= 000014           UIPDR0= 177600          $HIBTS  001100          $TYPDS  005716
SIPDR4= 172210           TITL    006634           UIPDR1= 177602          $KTNEX  005362          $TYPE   006142
SIPDR5= 172212           TITLE   007755           UIPDR2= 177604          $KTOUT  005352          $TYPEC  006354
SIPDR6= 172214           TKVEC = 000060           UIPDR3= 177606          $KT11   005220          $TYPEX  006472
SIPDR7= 172216           TPVEC = 000064           UIPDR4= 177610          $LF     006506          $TYPOC  005514
SIZEHI= 177762           TRAPVE= 000034           UIPDR5= 177612          $LFLG   001413          $TYPON  005530
SIZELO= 177760           TRTVEC= 000014           UIPDR6= 177614          $LSTAD  005464          $TYPOS  005470
SR0    = 177572          TST1    002332           UIPDR7= 177616          $LSTBK  005466          $UNIT   001126
SR1    = 177574          TST10   003720           UPF     010055          $MADR1  001146          $UNITM  001110
SR2    = 177576          TST11   004052           USESTK= 000600          $MAIL   001114          $USWR   001140
SR3    = 172516          TST2    002554           USP    =%000006         $MAMS1  001144          $$GET4= 000000
SSP    =%000006          TST3    003004           UVEC    005040          $MBADR  001102          $OFILL  005713
STACK  = 001100          TST4    003264           VIR.AD  007716          $MFLG   001412          .$X   = 001100

. ABS.   010172      000
         000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  59304 WORDS  ( 232 PAGES)
DYNAMIC MEMORY:  20434 WORDS  ( 78 PAGES)
ELAPSED TIME:  00:03:33
CKKACA0,CKKACA0/NL:TOC/CRF/-SP=CKKACA0.SML,CKKACA0.P11
```

SYMBOL	VALUE	REFERENCES		
ABASE	= 000000	64-337		
ACDW1	= 000000	64-337		
ACDW2	= 000000	64-337		
ACPUOP	= 000000	64-337	64-337	
ADDW0	= 000000	64-337		
ADDW1	= 000000	64-337		
ADDW10	= 000000	64-337		
ADDW11	= 000000	64-337		
ADDW12	= 000000	64-337		
ADDW13	= 000000	64-337		
ADDW14	= 000000	64-337		
ADDW15	= 000000	64-337		
ADDW2	= 000000	64-337		
ADDW3	= 000000	64-337		
ADDW4	= 000000	64-337		
ADDW5	= 000000	64-337		
ADDW6	= 000000	64-337		
ADDW7	= 000000	64-337		
ADDW8	= 000000	64-337		
ADDW9	= 000000	64-337		
ADEVCT	= 000000	64-337	64-337	
ADEVM	= 000000	64-337		
AENV	= 000000	64-337	64-337	
AENVM	= 000000	64-337	64-337	
AFATAL	= 000000	64-337	64-337	
AMADR1	= 000000	64-337	64-337	
AMADR2	= 000000	64-337		
AMADR3	= 000000	64-337		
AMADR4	= 000000	64-337		
AMAMS1	= 000000	64-337	64-337	
AMAMS2	= 000000	64-337		
AMAMS3	= 000000	64-337		
AMAMS4	= 000000	64-337		
AMSGAD	= 000000	64-337	64-337	
AMSGLG	= 000000	64-337	64-337	
AMSGTY	= 000000	64-337	64-337	
AMTYP1	= 000000	64-337	64-337	
AMTYP2	= 000000	64-337		
AMTYP3	= 000000	64-337		
AMTYP4	= 000000	64-337		
APASS	= 000000	64-337	64-337	
APF	002134	64-394	#64-397	
APRIOR	= 000000	64-337		
APTCSU	= 000040	#64-338	74-816	
APTENV	= 000001	64-338	#64-338	74-816
APTSIZ	= 000200	#64-338	64-346	
APTSPO	= 000100	64-338	#64-338	74-816
ASWREG	= 000000	64-337	64-337	
ATESTN	= 000000	64-337	64-337	
AUNIT	= 000000	64-337	64-337	
AUSWR	= 000000	64-337	64-337	
AVECT1	= 000000	64-337		

SYMBOL	VALUE	REFERENCES								
AVECT2	= 000000	64-337								
BEGIN	001424	64-326	#64-346							
BIT0	= 000001	#64-325								
BIT00	= 000001	#64-325	64-325							
BIT01	= 000002	#64-325	64-325	71-734						
BIT02	= 000004	#64-325	64-325							
BIT03	= 000010	#64-325	64-325							
BIT04	= 000020	#64-325	64-325	73-804						
BIT05	= 000040	#64-325	64-325							
BIT06	= 000100	#64-325	64-325							
BIT07	= 000200	#64-325	64-325	64-355						
BIT08	= 000400	#64-325	64-325							
BIT09	= 001000	#64-325	64-325							
BIT1	= 000002	#64-325								
BIT10	= 002000	#64-325								
BIT11	= 004000	#64-325								
BIT12	= 010000	#64-325								
BIT13	= 020000	#64-325								
BIT14	= 040000	#64-325								
BIT15	= 100000	#64-325								
BIT2	= 000004	#64-325								
BIT3	= 000010	#64-325								
BIT4	= 000020	#64-325								
BIT5	= 000040	#64-325								
BIT6	= 000100	#64-325								
BIT7	= 000200	#64-325								
BIT8	= 000400	#64-325								
BIT9	= 001000	#64-325								
BOOT	010012	64-390	#79-1000							
BPTVEC	= 000014	#64-325								
CACHVE	= 000114	#64-325	64-348							
CHECK	004360	67-650	67-654	#70-682						
CONTRL	= 177746	#64-325								
CPUERR	= 177766	#64-325								
CR	= 000015	#64-325	74-816	74-816						
CRLF	= 000200	#64-325	64-395	64-397	64-398	64-400	74-816	74-816	74-855	74-855
		74-871	74-895							
DAT	006726	74-861	#74-871							
DATA	005056	69-663	70-684	#72-792						
DDISP	= 177570	#64-325	64-346							
DISPLA	= 177570	#64-343	*64-346	*64-346						
DISPRE	000174	#64-326	64-346							
DOWN	010125	72-780	#79-1003							
DSWR	= 177570	#64-325	64-346							
DVEC	005044	64-347	72-763	72-772	#72-788					
EMTVEC	= 000030	#64-325	*65-511	*65-512						
END	010170	69-665	70-686	71-726	#79-1010					
ERRAD	005036	*72-747	#72-785							
ERROR	= 104000	#64-325	64-420	64-432	64-443	65-472	65-483	65-494	65-525	65-536
		65-547	65-565	65-579	66-594	66-614	67-635	70-698	#74-833	
ERRPC	006564	#74-835	*74-854	74-874						
ERRVEC	= 000004	#64-325	64-346	64-346	64-346	65-560	65-567	65-573	65-582	66-589

SYMBOL	VALUE	REFERENCES								
		66-597	67-627	67-641	73-810	73-810	73-810	73-810	73-810	73-810
		73-810	73-810	73-810						
EXPDAT	007676	*70-696	70-701	77-952	77-975	#79-988				
EXP.AD	007702	*64-430	64-434	*65-492	65-496	*65-534	65-538	77-956	77-977	#79-990
EXP.PS	007712	*64-441	64-445	*65-481	65-485	*65-545	65-549	77-964	77-981	#79-994
EXP.SP	007706	*64-418	64-422	*65-470	65-474	*65-523	65-527	77-960	77-979	#79-992
FIN	007060	74-883	#74-893							
FLAG	005052	*66-588	*72-744	*72-745	#72-790					
FT	002264	64-396	#64-400	68-657						
GNS	= ******	64-326	64-326	64-392	64-395	64-397	64-398	64-400	64-410	65-460
		65-507	65-558	65-572	66-586	66-605	67-625	67-649	74-820	74-820
		74-820	74-820	74-820	74-820	74-820	74-820	74-820	74-820	74-855
		74-871	74-873	74-887	74-890	74-895	74-915	74-917	74-919	74-921
		74-923	74-925	74-927	75-930	75-932	77-936			
HIADRS	= 177742	#64-325	71-724							
HITMIS	= 177752	#64-325								
HT	= 000011	#64-325	74-816	74-816						
ILLDWN	005024	72-763	#72-779							
ILLUP	005012	72-746	#72-775							
IOTVEC	= 000020	#64-325								
KDPAR0	= 172360	#64-325								
KDPAR1	= 172362	#64-325								
KDPAR2	= 172364	#64-325								
KDPAR3	= 172366	#64-325								
KDPAR4	= 172370	#64-325								
KDPAR5	= 172372	#64-325								
KDPAR6	= 172374	#64-325								
KDPAR7	= 172376	#64-325								
KDPDR0	= 172320	#64-325								
KDPDR1	= 172322	#64-325								
KDPDR2	= 172324	#64-325								
KDPDR3	= 172326	#64-325								
KDPDR4	= 172330	#64-325								
KDPDR5	= 172332	#64-325								
KDPDR6	= 172334	#64-325								
KDPDR7	= 172336	#64-325								
KERSTK	= 001100	#64-325								
KIPAR0	= 172340	#64-325	73-796	73-810	73-810					
KIPAR1	= 172342	#64-325								
KIPAR2	= 172344	#64-325								
KIPAR3	= 172346	#64-325								
KIPAR4	= 172350	#64-325								
KIPAR5	= 172352	#64-325	69-673	70-695	70-710	73-798				
KIPAR6	= 172354	#64-325	73-800							
KIPAR7	= 172356	#64-325	73-802	73-810						
KIPDR0	= 172300	#64-325	73-797							
KIPDR1	= 172302	#64-325								
KIPDR2	= 172304	#64-325								
KIPDR3	= 172306	#64-325								
KIPDR4	= 172310	#64-325								
KIPDR5	= 172312	#64-325	73-799							
KIPDR6	= 172314	#64-325	73-801							

SYMBOL	VALUE	REFERENCES					
KIPDR7	= 172316	#64-325	73-803				
KSP	=%000006	#64-325					
LF	= 000012	#64-325	64-395	64-397	64-398	74-816	74-816
LIMIT	005054	*64-362	69-662	70-683	#72-791		
LKS	= 177546	#64-325					
LKVEC	= 000100	#64-325					
LOAD	004254	67-647	#69-661				
LOADRS	= 177740	#64-325	71-726				
LRTI	001422	#64-344	65-511	65-532	65-534		
MAINT	= 177750	#64-325					
MAP	005060	67-628	69-661	70-682	#73-796		
MAPH0	= 170202	#64-325					
MAPH00	= 170202	#64-325	64-325				
MAPH01	= 170206	#64-325	64-325				
MAPH02	= 170212	#64-325	64-325				
MAPH03	= 170216	#64-325	64-325				
MAPH04	= 170222	#64-325	64-325				
MAPH05	= 170226	#64-325	64-325				
MAPH06	= 170232	#64-325	64-325				
MAPH07	= 170236	#64-325	64-325				
MAPH1	= 170206	#64-325					
MAPH10	= 170242	#64-325					
MAPH11	= 170246	#64-325					
MAPH12	= 170252	#64-325					
MAPH13	= 170256	#64-325					
MAPH14	= 170262	#64-325					
MAPH15	= 170266	#64-325					
MAPH16	= 170272	#64-325					
MAPH17	= 170276	#64-325					
MAPH2	= 170212	#64-325					
MAPH20	= 170302	#64-325					
MAPH21	= 170306	#64-325					
MAPH22	= 170312	#64-325					
MAPH23	= 170316	#64-325					
MAPH24	= 170320	#64-325					
MAPH25	= 170326	#64-325					
MAPH26	= 170332	#64-325					
MAPH27	= 170336	#64-325					
MAPH3	= 170216	#64-325					
MAPH30	= 170342	#64-325					
MAPH31	= 170346	#64-325					
MAPH32	= 170352	#64-325					
MAPH33	= 170356	#64-325					
MAPH34	= 170362	#64-325					
MAPH35	= 170366	#64-325					
MAPH36	= 170372	#64-325					
MAPH37	= 170376	#64-325					
MAPH4	= 170222	#64-325					
MAPH5	= 170226	#64-325					
MAPH6	= 170232	#64-325					
MAPH7	= 170236	#64-325					
MAPL0	= 170200	#64-325					

SYMBOL	VALUE	REFERENCES								
MAPL00	= 170200	#64-325	64-325							
MAPL01	= 170204	#64-325	64-325							
MAPL02	= 170210	#64-325	64-325							
MAPL03	= 170214	#64-325	64-325							
MAPL04	= 170220	#64-325	64-325							
MAPL05	= 170224	#64-325	64-325							
MAPL06	= 170230	#64-325	64-325							
MAPL07	= 170234	#64-325	64-325							
MAPL1	= 170204	#64-325								
MAPL10	= 170240	#64-325								
MAPL11	= 170244	#64-325								
MAPL12	= 170250	#64-325								
MAPL13	= 170254	#64-325								
MAPL14	= 170260	#64-325								
MAPL15	= 170264	#64-325								
MAPL16	= 170270	#64-325								
MAPL17	= 170274	#64-325								
MAPL2	= 170210	#64-325								
MAPL20	= 170300	#64-325								
MAPL21	= 170304	#64-325								
MAPL22	= 170310	#64-325								
MAPL23	= 170314	#64-325								
MAPL24	= 170320	#64-325								
MAPL25	= 170324	#64-325								
MAPL26	= 170330	#64-325								
MAPL27	= 170334	#64-325								
MAPL3	= 170214	#64-325								
MAPL30	= 170340	#64-325								
MAPL31	= 170344	#64-325								
MAPL32	= 170350	#64-325								
MAPL33	= 170354	#64-325								
MAPL34	= 170360	#64-325								
MAPL35	= 170364	#64-325								
MAPL36	= 170370	#64-325								
MAPL37	= 170374	#64-325								
MAPL4	= 170220	#64-325								
MAPL5	= 170224	#64-325								
MAPL6	= 170230	#64-325								
MAPL7	= 170234	#64-325								
MEMERR	= 177744	#64-325	71-731	71-737	71-737					
MMERR	005156	73-805	#73-807	73-808						
MMR0	= 177572	#64-325	64-325	67-632	67-639	69-667	69-674	69-676	70-688	70-711
		70-713								
MMR1	= 177574	#64-325	64-325							
MMR2	= 177576	#64-325	64-325							
MMR3	= 172516	#64-325	64-325	73-804						
MMVEC	= 000250	#64-325	67-629	69-664	70-685	73-805				
PARERR	004554	64-348	#71-719							
PARFLG	004556	64-349	#71-720	*71-738						
PAR.OF	007720	*70-695	70-700	77-970	77-984	#79-997				
PFAIL	001416	#64-340	*64-369	64-372	*64-378	64-393	64-413	65-464	65-513	65-562
		65-575	66-592	66-609	67-631	67-651				

SYMBOL	VALUE	REFERENCES								
PIRQ	= 177772	#64-325								
PIRQVE	= 000240	#64-325								
PNAME	007722	#79-998								
POWDWN	004654	64-347	#72-744	72-772						
POWUP	004750	72-759	#72-763							
PRDATA	007506	74-881	#77-942							
PRETST	002274	64-386	#64-401							
PRTABL	007652	74-858	74-880	#77-974						
PRTITL	007126	74-859	#74-905							
PR0	= 000000	#64-325								
PR1	= 000040	#64-325								
PR2	= 000100	#64-325								
PR3	= 000140	#64-325								
PR4	= 000200	#64-325								
PR5	= 000240	#64-325								
PR6	= 000300	#64-325								
PR7	= 000340	#64-325								
PS	= 177776	#64-325	64-325	64-411	*65-462	65-508	65-559	65-577	66-587	66-606
		67-626	67-646							
PSW	= 177776	#64-325								
PWRVEC	= 000024	#64-325								
RECDAT	007700	*70-697	70-702	77-954	77-976	#79-989				
REC.AD	007704	*64-431	64-435	*65-493	65-497	*65-535	65-539	77-958	77-978	#79-991
REC.PS	007714	*64-442	64-446	*65-482	65-486	65-546	65-550	77-966	77-982	#79-995
REC.SP	007710	*64-419	64-423	*65-471	65-475	*65-524	65-528	77-962	77-980	#79-993
RESVEC	= 000010	#64-325								
R10	=%000000	#64-325								
R11	=%000001	#64-325								
R12	=%000002	#64-325								
R13	=%000003	#64-325								
R14	=%000004	#64-325								
R15	=%000005	#64-325								
R6	=%000006	#64-325								
R7	=%000007	#64-325								
SAVR0	006620	*74-838	#74-845	74-897						
SAVR1	006622	*74-839	#74-846	74-898						
SAVR2	006624	*74-840	#74-847	74-899						
SAVR3	006626	*74-841	#74-848	74-900						
SAVR4	006630	*74-842	#74-849	74-901						
SAVR5	006632	*74-843	#74-850	74-902						
SAV6	005050	*72-757	72-764	#72-789						
SCOPE	= 000004	#64-325								
SDPAR0	= 172260	#64-325								
SDPAR1	= 172262	#64-325								
SDPAR2	= 172264	#64-325								
SDPAR3	= 172266	#64-325								
SDPAR4	= 172270	#64-325								
SDPAR5	= 172272	#64-325								
SDPAR6	= 172274	#64-325								
SDPAR7	= 172276	#64-325								
SDPDR0	= 172220	#64-325								
SDPDR1	= 172222	#64-325								

SYMBOL	VALUE	REFERENCES								
SDPDR2	= 172224	#64-325								
SDPDR3	= 172226	#64-325								
SDPDR4	= 172230	#64-325								
SDPDR5	= 172232	#64-325								
SDPDR6	= 172234	#64-325								
SDPDR7	= 172236	#64-325								
SIPAR0	= 172240	#64-325	64-406	72-758	72-765					
SIPAR1	= 172242	#64-325	64-407							
SIPAR2	= 172244	#64-325	64-408							
SIPAR3	= 172246	#64-325	64-409							
SIPAR4	= 172250	#64-325								
SIPAR5	= 172252	#64-325								
SIPAR6	= 172254	#64-325								
SIPAR7	= 172256	#64-325								
SIPDR0	= 172200	#64-325								
SIPDR1	= 172202	#64-325								
SIPDR2	= 172204	#64-325								
SIPDR3	= 172206	#64-325								
SIPDR4	= 172210	#64-325								
SIPDR5	= 172212	#64-325								
SIPDR6	= 172214	#64-325								
SIPDR7	= 172216	#64-325								
SIZEHI	= 177762	#64-325								
SIZELO	= 177760	#64-325								
SR0	= 177572	#64-325	73-810	73-810	73-810					
SR1	= 177574	#64-325								
SR2	= 177576	#64-325								
SR3	= 172516	#64-325	73-810							
SSP	=%000006	#64-325								
STACK	= 001100	#64-325	64-325	64-325	64-325	64-346	64-416	64-426	64-427	64-438
		65-461	65-467	65-468	65-478	65-489	65-509	65-517	65-519	65-521
		65-531	65-542	65-563	65-576	65-581	66-608	66-617	67-633	67-640
		67-653	72-750							
STACK1	= 000500	#64-327	64-420	64-432	64-443	65-472	65-483	65-494	65-525	65-536
		65-547	65-565	65-579	66-594	66-614	67-635	70-698		
STKLMT	= 177774	#64-325								
SUPSTK	= 000700	#64-325								
SWR	= 177570	#64-342	64-343	*64-346	64-346	*64-346	*64-346	64-449	65-500	65-553
		65-568	65-583	66-601	66-618	67-642	67-655			
SWREG	000176	#64-326	64-346							
SW0	= 000001	#64-325								
SW00	= 000001	#64-325	64-325							
SW01	= 000002	#64-325	64-325							
SW02	= 000004	#64-325	64-325							
SW03	= 000010	#64-325	64-325							
SW04	= 000020	#64-325	64-325							
SW05	= 000040	#64-325	64-325							
SW06	= 000100	#64-325	64-325							
SW07	= 000200	#64-325	64-325							
SW08	= 000400	#64-325	64-325							
SW09	= 001000	#64-325	64-325							
SW1	= 000002	#64-325								

SYMBOL	VALUE	REFERENCES
SW10	= 002000	#64-325
SW11	= 004000	#64-325
SW12	= 010000	#64-325
SW13	= 020000	#64-325
SW14	= 040000	#64-325 64-449 65-500 65-553 65-568 65-583 66-601 66-618 67-642 67-655
SW15	= 100000	#64-325
SW2	= 000004	#64-325
SW3	= 000010	#64-325
SW4	= 000020	#64-325
SW5	= 000040	#64-325
SW6	= 000100	#64-325
SW7	= 000200	#64-325
SW8	= 000400	#64-325
SW9	= 001000	#64-325
SYSTID	= 177764	#64-325
TBITVE	= 000014	#64-325
TITL	006634	74-844 #74-852
TITLE	007755	64-388 #79-999
TKVEC	= 000060	#64-325
TPVEC	= 000064	#64-325
TRAPVE	= 000034	#64-325 64-346 64-346
TRTVEC	= 000014	#64-325
TST1	002332	#64-410 64-449
TST10	003720	#67-625 67-642
TST11	004052	#67-645 67-655
TST2	002554	#65-460 65-500
TST3	003004	#65-507 65-553
TST4	003264	#65-558 65-568
TST5	003370	#65-572 65-583
TST6	003500	#66-586 66-601
TST7	003620	#66-605 66-618
TYPDS	= 104405	64-391 68-657 #74-820
TYPE	= 104401	64-387 64-389 64-392 64-395 64-397 64-398 64-400 64-410 64-410 65-460 65-460 65-507 65-507 65-558 65-558 65-572 65-572 66-586 66-586 66-605 66-605 67-625 67-625 67-649 67-649 68-657 68-657 72-775 72-779 74-814 74-815 74-816 #74-820 74-855 74-871 74-873 74-875 74-887 74-889 74-890 74-895 74-915 74-917 74-919 74-921 74-923 74-925 74-927 75-930 75-932 77-936
TYPOC	= 104402	#74-820 74-874 77-952 77-954 77-956 77-958 77-960 77-962 77-964 77-966 77-968 77-970
TYPON	= 104404	#74-820
TYPOS	= 104403	#74-820 74-872
UDPAR0	= 177660	#64-325
UDPAR1	= 177662	#64-325
UDPAR2	= 177664	#64-325
UDPAR3	= 177666	#64-325
UDPAR4	= 177670	#64-325
UDPAR5	= 177672	#64-325
UDPAR6	= 177674	#64-325
UDPAR7	= 177676	#64-325
UDPDR0	= 177620	#64-325

SYMBOL	VALUE	REFERENCES							
UDPDR1	= 177622	#64-325							
UDPDR2	= 177624	#64-325							
UDPDR3	= 177626	#64-325							
UDPDR4	= 177630	#64-325							
UDPDR5	= 177632	#64-325							
UDPDR6	= 177634	#64-325							
UDPDR7	= 177636	#64-325							
UIPAR0	= 177640	#64-325							
UIPAR1	= 177642	#64-325							
UIPAR2	= 177644	#64-325							
UIPAR3	= 177646	#64-325							
UIPAR4	= 177650	#64-325							
UIPAR5	= 177652	#64-325							
UIPAR6	= 177654	#64-325							
UIPAR7	= 177656	#64-325							
UIPDR0	= 177600	#64-325							
UIPDR1	= 177602	#64-325							
UIPDR2	= 177604	#64-325							
UIPDR3	= 177606	#64-325							
UIPDR4	= 177610	#64-325							
UIPDR5	= 177612	#64-325							
UIPDR6	= 177614	#64-325							
UIPDR7	= 177616	#64-325							
UPF	010055	72-776	#79-1001						
USESTK	= 000600	#64-325							
USP	=%000006	#64-325							
UVEC	005040	72-746	72-759	#72-787					
VIR.AD	007716	*70-692	*70-693	*70-694	70-699	77-968	77-983	#79-996	
XYZ	001420	#64-341	64-378	64-393					
$APTHD	001100	64-336	#64-336						
$ASTAT	= ******	64-338	64-338						
$ATYC	001174	64-338	#64-338						
$ATY1	001150	#64-338							
$ATY3	001156	#64-338	74-816						
$ATY4	001166	#64-338							
$CHARC	006470	*74-816	*74-816	74-816	*74-816	#74-816			
$CKSWR	= ******	74-820							
$CMTAG	= ******	64-346	64-346						
$CORE	005370	73-810	#73-810						
$CPUOP	001142	#64-337							
$CRLF	006505	74-816	74-816	74-816	#74-816				
$CROUT	005420	73-810	#73-810						
$DBLK	006132	74-815	74-815	#74-815					
$DEVCT	001124	#64-337							
$DOAGN	004230	68-657	68-657	#68-657					
$DTBL	006122	74-815	#74-815						
$ENDAD	004220	64-334	64-385	#68-657					
$ENDCT	004166	64-346	#68-657						
$ENDMG	004237	68-657	#68-657						
$ENULL	004234	64-410	65-460	65-507	65-558	65-572	66-586	66-605	67-625	67-649
		68-657	#68-657	74-875	74-889					
$ENV	001134	#64-337	64-338	64-338	74-816					

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES								
$ENVM	001135	#64-337	64-338	64-346	74-816	74-816				
$EOP	004142	#68-657								
$EOPCT	004160	*64-346	#68-657	68-657						
$ERROR	006566	64-330	65-518	#74-838						
$ETABL	001134	#64-337								
$ETEND	001150	64-336	#64-337							
$FATAL	001116	#64-337	*64-338							
$FFLG	001414	*64-338	*64-338	64-338	*64-338	#64-338				
$FILLC	006502	74-816	74-816	74-816	#74-816					
$FILLS	006501	74-816	74-816	#74-816						
$GET42	004210	#68-657								
$GTSWR	= ******	74-820								
$HD	= 000000	64-289	64-289	64-289						
$HIBTS	001100	#64-336								
$KTNEX	005362	73-810	#73-810							
$KTOUT	005352	73-810	#73-810							
$KT11	005220	*64-355	#73-810	*73-810	*73-810					
$LF	006506	74-816	74-816	#74-816						
$LFLG	001413	*64-338	#64-338							
$LSTAD	005464	*73-810	#73-810							
$LSTBK	005466	*64-357	64-358	*73-810	#73-810					
$MADR1	001146	#64-337								
$MAIL	001114	64-336	64-336	#64-337	64-346	74-816				
$MAMS1	001144	#64-337								
$MBADR	001102	#64-336								
$MFLG	001412	*64-338	64-338	*64-338	#64-338					
$MSGAD	001130	#64-337	*64-338	64-338						
$MSGLG	001132	#64-337	*64-338							
$MSGTY	001114	#64-337	64-338	*64-338	64-338	*64-338				
$MTYP1	001145	#64-337								
$NULL	006500	74-816	74-816	74-816	#74-816					
$NWTST	= 000001	#64-410	64-410	#64-410	#65-460	65-460	#65-460	#65-507	65-507	#65-507
		#65-558	65-558	#65-558	#65-572	65-572	#65-572	#66-586	66-586	#66-586
		#66-605	66-605	#66-605	#67-625	67-625	#67-625	#67-645	67-645	#67-645
$OCNT	005712	*74-814	*74-814	#74-814						
$OMODE	005714	*74-814	*74-814	74-814	*74-814	*74-814	#74-814			
$PASS	001122	#64-337	*64-346	*64-346	*68-657	*68-657	68-657	68-657	68-657	
$PASTM	001106	#64-336								
$QUES	006504	74-816	74-816	#74-816						
$RDCHR	= ******	74-820								
$RDDEC	= ******	74-820								
$RDLIN	= ******	74-820								
$RDOCT	= ******	74-820								
$RTNAD	004232	#68-657								
$R2A	= ******	74-820								
$SAVPC	= 000204	#64-328	64-332							
$SAVRE	= ******	74-820								
$SETUP	= 000024	#64-333	64-333	#64-333	64-333	#64-333	64-346	64-346	64-346	64-346
		64-346	64-346	64-346	64-346	64-346	64-346	64-346	64-346	68-657
		68-657								
$SIZE	005162	64-356	#73-810							
$SIZEX	005424	73-810	#73-810							

SYMBOL CROSS REFERENCE CREF

SYMBOL	VALUE	REFERENCES								
$STUP	= 177777	#64-333	#64-333	64-333	#64-333	#64-333	64-333			
$SVPC	= 000204	#64-334	64-334							
$SWR	= 040000	#64-286	64-289	64-292	64-292	64-292	64-292	64-292	64-292	64-292
		64-292	64-346	64-410	65-460	65-507	65-558	65-572	66-586	66-605
		67-625	67-645	68-657	68-657	68-657	68-657	68-657		
$SWREG	001136	#64-337	64-346							
$TESTN	001120	#64-337	*64-401	*65-459	*65-506	*65-557	*65-571	*66-585	*66-604	*67-624
		*67-644	74-872							
$TKB	006512	74-816	74-816	#74-818						
$TKS	006510	74-816	74-816	#74-817						
$TN	= 000012	#64-287	64-289	64-410	64-410	64-410	#64-410	64-449	65-460	65-460
		65-460	#65-460	65-500	65-507	65-507	65-507	#65-507	65-553	65-558
		65-558	65-558	#65-558	*65-568	65-572	65-572	65-572	#65-572	*65-583
		66-586	66-586	66-586	#66-586	66-601	66-605	66-605	66-605	#66-605
		*66-618	67-625	67-625	67-625	#67-625	*67-642	67-645	67-645	#67-645
		*67-649	67-655							
$TPB	006476	74-816	74-816	74-816	#74-816					
$TPFLG	006503	74-816	74-816	74-816	#74-816					
$TPS	006474	74-816	74-816	74-816	#74-816					
$TRAP	006514	64-346	#74-820							
$TRAP2	006536	#74-820	74-820							
$TRP	= 000006	#74-820	74-820	74-820	74-820	74-820	#74-820	74-820	74-820	74-820
		74-820	#74-820	74-820	74-820	74-820	74-820	#74-820	74-820	74-820
		74-820	74-820	#74-820	74-820	74-820	74-820	74-820	#74-820	
$TRPAD	006550	74-820	#74-820							
$TSTM	001104	#64-336								
$TYPBN	= ******	74-820								
$TYPDS	005716	#74-815	74-820	74-820						
$TYPE	006142	64-338	#74-816	74-820	74-820					
$TYPEC	006354	74-816	74-816	74-816	#74-816	74-816				
$TYPEX	006472	74-816	74-816	#74-816						
$TYPOC	005514	#74-814	74-820	74-820						
$TYPON	005530	74-814	#74-814	74-820						
$TYPOS	005470	#74-814	74-820							
$UNIT	001126	#64-337								
$UNITM	001110	#64-336								
$USWR	001140	#64-337								
$$GET4	= 000000	#68-657	68-657							
$OFILL	005713	*74-814	*74-814	74-814	#74-814					
.$ASTA	= ******	64-338	64-338							
.$X	= 001100	#64-336	64-336							

MACRO NAME	REFERENCES									
COMMEN	#18-1530	#64-325								
ENDCOM	#18-1542	#64-325								
ERR	#64-300	#64-420	#64-432	#64-443	#65-472	#65-483	#65-494	#65-525	#65-536	#65-547
	#65-565	#65-579	#66-594	#66-614	#67-635	#70-698				
ESCAPE	#20-1658	#64-325								
GETPRI	#13-1282	#64-325	73-810							
GETSWR	#22-1729	#64-325								
LOOP	#64-318	64-449	65-500	65-553	65-568	65-583	66-601	66-618	67-642	67-655
LOOPX	#64-321	#64-449	#65-500	#65-553	#65-568	#65-583	#66-601	#66-618	#67-642	#67-655
MULT	#43-4409	#64-325								
NEWTST	#19-1589	#64-325	#64-410	#65-460	#65-507	#65-558	#65-572	#66-586	#66-605	#67-625
	#67-645									
PFAIL	#64-294	64-413	65-464	65-513	65-562	65-575	66-592	66-609	67-631	67-651
POP	#26-2107	#64-325	64-338	64-338	74-815					
PUSH	#26-2099	#64-325	64-338	64-338	64-338	74-815				
REPORT	#56-5368	#64-325								
SETK	#64-315	64-411	65-508	65-559	65-577	66-587	66-606	67-626	67-646	
SETPRI	#12-1250	#64-325								
SETTRA	#74-820	74-820	74-820	74-820	74-820	74-820				
SETUP	#14-1306	#64-325	#64-346							
SKIP	#21-1692	#64-325								
SLASH	#17-1482	#64-325								
SPACE	#64-297	#64-325								
STARS	#16-1451	#64-325	64-334	64-336	64-336	64-336	64-337	64-338	64-410	64-410
	65-460	65-460	65-507	65-507	65-558	65-558	65-572	65-572	66-586	66-586
	66-605	66-605	67-625	67-625	67-645	67-645	68-657	73-810	74-814	74-815
	74-816	74-820								
SWRSU	#15-1420	#64-325	#64-346	#64-346						
TEST	#64-308	64-410	65-460	65-507	65-558	65-572	66-586	66-605	67-625	67-649
TRMTRP	#74-820									
TSTNUM	#64-311	64-410	65-460	65-507	65-558	65-572	66-586	66-605	67-625	67-649
TYPBIN	#25-2043	#64-325								
TYPDEC	#25-2013	#64-325	#64-391	#68-657						
TYPNAM	#23-1783	#64-325								
TYPNUM	#25-1980	#64-325								
TYPOCS	#25-1933	#64-325	#74-872							
TYPOCT	#25-1896	#64-325	#74-874	#77-952	#77-954	#77-956	#77-958	#77-960	#77-962	#77-964
	#77-966	#77-968	#77-970							
TYPTXT	#24-1850	#64-325	64-392	64-395	64-397	64-398	64-400	64-410	65-460	65-507
	65-558	65-572	66-586	66-605	67-625	67-649	74-855	74-871	74-873	74-887
	74-890	74-895	74-915	74-917	74-919	74-921	74-923	74-925	74-927	75-930
	75-932	77-936								
$$ESCA	#20-1671	#64-325								
$$NEWT	#19-1625	#64-325	#64-410	#65-460	#65-507	#65-558	#65-572	#66-586	#66-605	#67-625
	#67-645									
$$SET	#74-820	#74-820	#74-820	#74-820	#74-820	#74-820				
$$SETM	#64-346	#64-346								
$$SKIP	#21-1705	#64-325								
.EQUAT	#4-180									
.EQUIV	#64-280	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325
	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325
	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325

MACRO NAME	REFERENCES									
	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325
	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325	#64-325
	#64-325	#64-325	#64-325							
.HEADE	#2-54	#64-283	64-289							
.KT11	#5-323									
.SETUP	#10-1184	#64-283	#64-284	#64-333						
.SWRHI	#3-96	#64-283	64-292							
.SWRLO	#64-292									
.$ACT1	#52-4977	#64-284	#64-334							
.$APTB	#53-5021	#64-284	64-337							
.$APTH	#54-5277	#64-284	64-336							
.$APTY	#57-5452	#64-284	#64-338							
.$ASTA	#55-5323									
.$CATC	#7-909	#64-283	64-326							
.$CMTA	#9-1020									
.$DB2D	#46-4607									
.$DB20	#48-4730									
.$DIV	#45-4510									
.$EOP	#27-2166	#64-283	#68-657							
.$ERRO	#29-2647									
.$ERRT	#30-2842									
.$MULT	#44-4447									
.$POWE	#40-4159	#64-283								
.$RAND	#41-4234									
.$RDDE	#37-3830									
.$RDOC	#36-3739									
.$READ	#35-3344									
.$R2AZ	#51-4874									
.$SAVE	#38-3905									
.$SB2D	#47-4691									
.$SB20	#49-4792									
.$SCOP	#28-2401									
.$SIZE	#42-4287	#64-284	#73-810							
.$SUPR	#50-4830									
.$TRAP	#39-4007	#64-284	#74-820							
.$TYPB	#34-3237									
.$TYPD	#33-3160	#64-284	#74-815							
.$TYPE	#31-2929	#64-283	74-816							
.$TYPO	#32-3064	#64-283	#74-814							
.$40CA	#8-948									
.1170	#6-502	#64-283	#64-325							