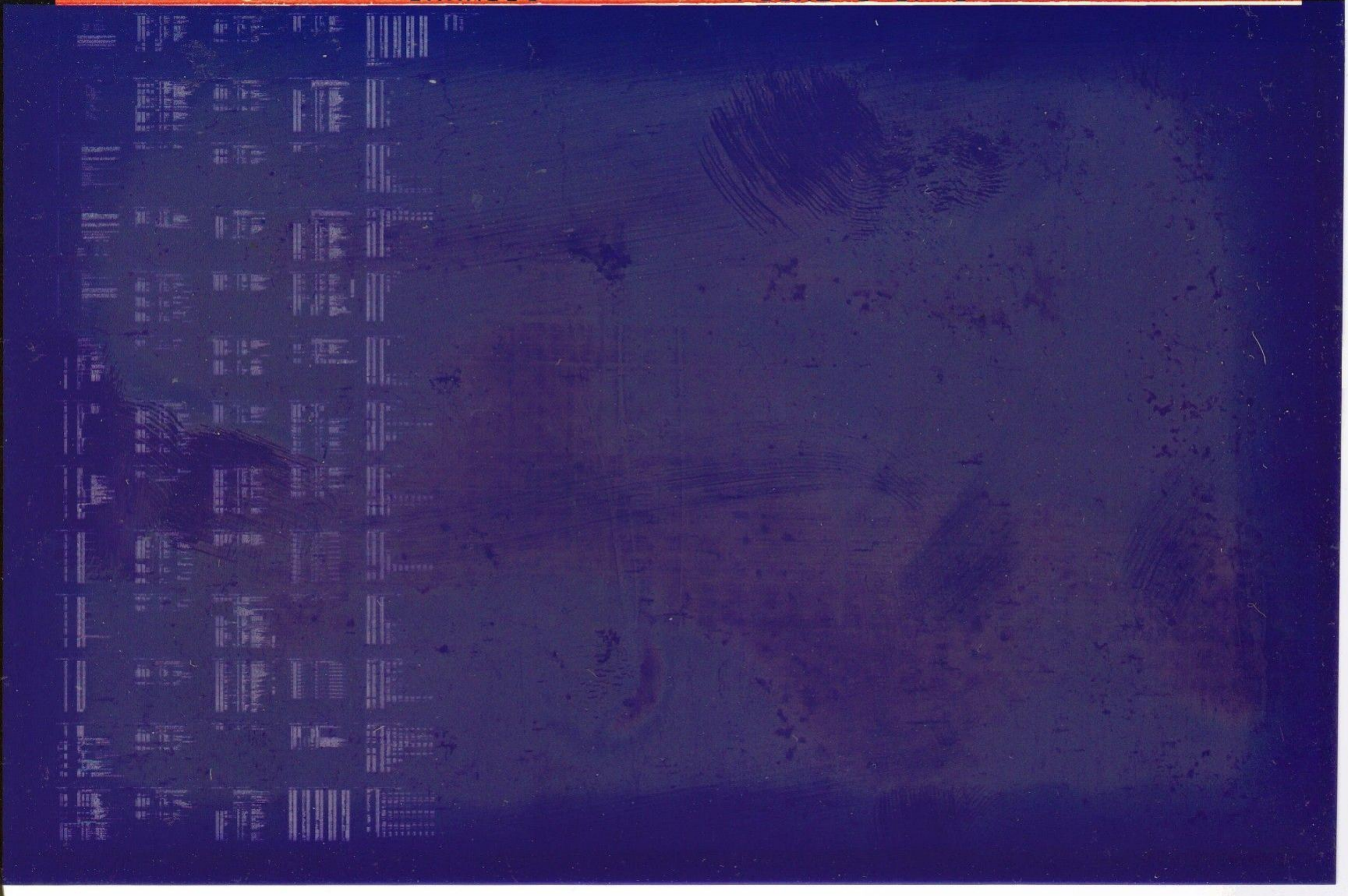
KD11-Z 11/44 POWER FAIL AH-F617C-MC AUG 1981
CKKACCO FICHE 1 OF 1 COPYRIGHT© 79-81
MADE IN USA



```
1                       .REM    %
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
```

IDENTIFICATION

PRODUCT CODE:	AC-F615C-MC
PRODUCT NAME:	CKKACCO 11/44 POWER FAIL
DATE CREATED:	APRIL, 1981
MAINTAINER:	DIAGNOSTIC ENGINEERING
AUTHORS:	DAN MILLEVILLE

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLED BY DIGITAL.

COPYRIGHT (C) 1979, 1981 BY DIGITAL EQUIPMENT CORPORATION

38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81

CONTENTS

1. ABSTRACT

2. REQUIREMENTS

2.1 EQUIPMENT

2.2 STORAGE

2.3 PRELIMINARY PROGRAMS

3. LOADING PROCEDURE

4. STARTING PROCEDURE

4.1 STARTING ADDRESS

4.2 CONTROL SWITCH SETTINGS

5. OPERATING PROCEDURE

5.1 MANUAL

5.2 AUTOMATIC

5.3 APT SETUP

5.4 SUBROUTINE ABSTRACT

6. ERRORS

7. RESTRICTIONS

8. MISCELLANEOUS

8.1 EXECUTION TIME

8.2 STACK POINTER

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

PDP 11/44 STANDARD COMPUTER WITH UP TO 1M WORDS OF CORE MEMORY. 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.

5. OPERATING PROCEDURE

5.1 MANUAL

START PROGRAM AT ADDRESS 200.
A MESSAGE WILL BE TYPED 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 THE POWER DOWN AND UP FOR EACH TEST UNTIL THE END OF PASS MESSAGE IS RECEIVED. RANDOM CHARACTERS (INCLUDING FORM FEED) MAY PRINT DURING POWER CYCLING. THIS IS ACCEPTABLE.

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. RANDOM CHARACTERS (INCLUDING FORM FEED) MAY PRINT DURING POWER CYCLING. THIS IS ACCEPTABLE.

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 PERTINENT INFOR-
                        MATION 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 SWITCH 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.

                6.3     ERROR RECOVERY

                        CONTINUE OR RESTART AT 200

                7.      RESTRICTIONS

                        NONE

                8.      MISCELLANEOUS

                8.1     EXECUTION TIME

                        5 SEC IF IN AUTOMATIC , N/A IF IN MANUAL

                8.2     STACK POINTER

                        STACK IS INITALLY SET TO 1100
                %
```

```
                        .TITLE  CKKACC0 11/44 POWER FAIL
                        ;*COPYRIGHT (C) 1981
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DAN MILLEVILLE
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C5), JAN, 1981.
                        ;*


                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        14            LOOP ON TEST
                        ;*        10            INHIBIT BELL AT END-OF-PASS

                        .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                  MAPL0C = 170200
170202                  MAPH0C = 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
```

UNIBUS MAP REGISTER DEFINITIONS

```
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
  280                                   .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
  282           000500                  STACK1=500
  283           000204                  $SAVPC=.
  284           000030                  .=30
  285 000030    006550                  $ERROR
  286 000032    000340                  340
  287           000204                  .=$SAVPC
  289                                   .SBTTL  ACT11 HOOKS
                                        ;;*****************************************************************
                                        ;HOOKS REQUIRED BY ACT11
                000204                          $SVPC=.                 ;SAVE PC
                000046                          .=46
     000046     003724                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                          .=52
     000052     000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                000204                          .=$SVPC                 ;; RESTORE PC
  290           001100                  .=1100
  291                                   .SBTTL  APT PARAMETER BLOCK
                                        ;;*****************************************************************
                                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                        ;;*****************************************************************
                001100                          .$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)
292                             .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
293                             .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  004566                  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
294
295 001416  000000                      PFAIL: .WORD 0
296 001420  000000                      XYZ: .WORD 0
297         177570                      SWR=177570
298         177570                      DISPLAY=SWR
299 001422  000002                      LRTI:   RTI
```

```
    500 001424                                  BEGIN:
                                                .SBTTL  INITIALIZE THE COMMON TAGS
        001424  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                                ;;INITIALIZE A FEW VECTORS
        001430  012737  006476  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  002216  002206                  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     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
        001566                          67$:
    301 001566  012777  004360  002760                  MOV     #POWDWN,@DVEC   ;SET UP POWER DOWN VECTOR
    302 001574  012737  004260  000114                  MOV     #PARERR,@#CACHVEC ;SET UP PARITY ERROR VECTOR
    303 001602  012737  000001  004262                  MOV     #1,@#PARFLG     ;INITIALIZE THE MULTI PARITY ERROR INDICATOR
    304
    305                                         ;DETERMINE THE SIZE OF MEMORY IN SYSTEM USING SYSMAC SIZE
    306                                         ;ROUTINE
    307
    308 001610  005000                                  CLR     R0              ;MAKE MSB'S = 0
    309 001612  052767  000200  003170                  BIS     #BIT07,$KT11    ;TURN ON MEMORY MANAGMENT
    310 001620  004767  003100                          JSR     PC,$SIZE        ;GET PAR VALUE FOR LAST WORD
    311 001624  062767  000037  003604                  ADD     #37,$LSTBK      ;FINE LAST ADDRESS OF MEMORY BANK
    312 001632  016701  003600                          MOV     $LSTBK,R1       ;CHECK LAST BANK WITH R1
    313 001636  005201                                  INC     R1              ;SIZE MEMORY
    314 001640  071027  000200                          DIV     #200,R0         ;FORM THE BLOCK NUMBER OF
    315 001644  005300                                  DEC     R0              ; THE LAST ADDRESS AND
    316 001646  010067  002712                          MOV     R0,LIMIT        ;SAVE IT.
    317 001652  005200                                  INC     R0              ;MULT BY 4
    318 001654  006300                                  ASL     R0
    319 001656  006300                                  ASL     R0
    320
    321
    322                                         ;THIS ROUTINE SIZE FOR PFAIL HARDWARE,TO AUTOMATIC POWER FAIL THE SYSTEM
    323 001660  012767  177740  177530                  MOV     #177740,PFAIL   ;SETUP PFAIL ADDRESS
    324 001666  012767  001710  176110                  MOV     #3$,4           ;SET PC
    325 001674  012767  000340  176104                  MOV     #340,6          ;SET PSW
    326 001702  005777  177510                          TST     @PFAIL          ;READ PFAIL
    327 001706  000412                                  BR      4$              ;POWER FAIL HARDWARE IS IN SYSTEM
    328 001710  012767  000006  176066  3$:            MOV     #6,4            ;TRAP RETURN
    329 001716  005067  176064                          CLR     6
```

```
;INITIALIZE THE COMMON TAGS

    330 001722  005726                          TST     (SP)+           ;SETUP STACK PC
    331 001724  005726                          TST     (SP)+           ;SETUP STACK PSW
    332 001726  012767  001420  177462          MOV     #XYZ,PFAIL      ;SET UP DUMMY ADDRESS
    333 001734  012767  000006  176042  4$:     MOV     #6,4            ;RESTORE TRAP VEC
    334 001742  005067  176040                  CLR     6
```

INITIALIZE THE COMMON TAGS

```
                                        ;IDENTIFY THE PROGRAM AND TYPE INSTRUCTIONS
                                        
001746  023767  000042  001750          CMP     @#42,$ENDAD
001754  001421                          BEQ     PRETST
001756  104401  007737                  TYPE    ,TITLE          ;TYPE THE TITLE
001762  104401  007774                  TYPE    ,BOOT
001766  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
001770  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
001772  104401  010151                  TYPE    ,SUPMSG         ;TYPE THE STARTUP MESSAGE
001776  026727  177414  001420          CMP     PFAIL,#XYZ      ;IS AUTOMATIC POWER FAIL
002004  001403                          BEQ     5$              ;BRANCH IF NO
002006  104401  010371                  TYPE    ,HDWPFL         ;TYPE HARDWARE POWER FAILURE MESSAGE
002012  000402                          BR      PRETST          ;BRANCH OVER MANUAL MESSAGE
002014  104401  010311          5$:     TYPE    ,USRPFL         ;TYPE USER POWER FAIL MESSAGE
002020  012767  000001  177072  PRETST: MOV     #1,$TESTN
002026  104401  006473                  TYPE    ,$CRLF          ;TYPE A <CRLF>
                                        ;THIS ROUTINE IS FOR CHECKING 2MS TIMING USING SOB OUT OF SIPARS
                                        ;BRANCH . OUT OF SIPAR0=1.2US
                                        ;SOB HAS ONE MORE MICRO STEP THROUGH EQUAL 1.4US
                                        ;1024 MUL BY 1.4US EQUAL 1.43MS MIN TIME CHECKED FOR ILLUP,ILLDWN
002032  004767  002634                  JSR     PC,SETPAR       ;GO PUT TIME FACTOR SUBROUTINE IN SIPAR0-SIPAR3
```

```
                        .SBTTL  TEST # 1 - SIMPLE DOWN/UP TEST (KERNEL)
                        ;;*****************************************************************
                        ;*TEST 1         SIMPLE DOWN/UP TEST (KERNEL)
                        ;;*****************************************************************
                        TST1:
002036                          TYPE    ,65$            ;;TYPE ASCIZ STRING
002036  104401  002044        BR      64$             ;;GET OVER THE ASCIZ
002042  000402
                        ;;65$:  .ASCIZ  *1 *
002050                  64$:
002050  104401  003740        TYPE,$ENULL
356 002054  005037  177776      CLR     @#PS            ;SET KERNEL MODE
357 002060  012703  002074      MOV     #2$,R3          ;SET POWER UP RETURN
358 002064  012777  000001  17/324  MOV  #1,@PFAIL              ;SET UP TO START POWER FAIL
359 002072  000001              WAIT                    ,WAIT FOR POWER FAIL
360 002074  010600      2$:     MOV     SP,R0           ;GET SP
361 002076  022700  001074      CMP     #STACK-4,R0     ;CHECK SP
362 002102  001414              BEQ     3$              ;SKIP IF OK
363 002104  012767  001074  005556  MOV  #1074,EXP.SP1
364 002112  010067  005554      MOV     R0,REC.SP1
365 002116  012706  000500      MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                        ;FOR ERROR PRINTOUT
002122  104000                  ERROR

                                                                ;ERROR
                                                                ;-----
366                                                     ;SP NOT STACK-4
367 002124  007670              EXP.SP1                 ;PRINT EXPECTED STACK POINTER
368 002126  007672              REC.SP1                 ;PRINT  RECEIVED STACK POINTER
369 002130  000000              0
370 002132  000000              HALT
371 002134  012706  001100  3$:     MOV     #STACK,SP       ;RESET SP
372 002140  013700  001074      MOV     @#STACK-4,R0    ;GET RETURN ADDRESS
373 002144  022700  002074      CMP     #2$,R0          ;CHECK ADDRESS
374 002150  001414              BEQ     4$              ;SKIP IF OK
375 002152  012767  002074  005504  MOV  #2$,EXP.ADD
376 002160  010067  005502      MOV     R0,REC.ADD
377 002164  012706  000500      MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                        ;FOR ERROR PRINTOUT
002170  104000                  ERROR

                                                                ;ERROR
                                                                ;-----
378                                                     ;ADDRESS ON STACK IS WRONG
379 002172  007664              EXP.ADD                 ;PRINT EXPECTED ADDRESS
380 002174  007666              REC.ADD                 ;PRINT RECEIVED ADDRESS
381 002176  000000              0
382 002200  000000              HALT
383 002202  013700  001076  4$:     MOV     @#STACK-2,R0    ;GET OLD PS
384 002206  022700  000000      CMP     #0,R0           ;CHECK OLD PS
385 002212  001414              BEQ     5$              ;SKIP IF OK
386 002214  012767  000000  005452  MOV  #0,EXP.PSW
387 002222  010067  005450      MOV     R0,REC.PSW
388 002226  012706  000500      MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                        ;FOR ERROR PRINTOUT
002232  104000                  ERROR

                                                                ;ERROR
```

```
                                                                    ;-----
389
390 002234  007674                          EXP.PSW                ;OLD PS IS WRONG
391 002236  007676                          REC.PSW                ;PRINT EXPECTED PS
392 002240  000000                          0                      ;PRINT RECEIVED PS
393 002242  000000                          HALT
394 002244                          5$:
    002244  032737  040000  177570          BIT     #SW14,@#SWR    ;LOOP ON TEST?
    002252  001271                          BNE     TST1           ;LOOP TO TST1
395                                 ;
396                                 ;TEST ROUTINE TO CHECK RE-START CAPABILITY
397                                 ;USING THE BR. INSTRUCTION
398                                 ;OPERATOR MUST SET HALT SWITCH TO ENABLE POSITION
399                                 ;
```

```
    400                                 .SBTTL  TEST # 2 - POWER FAIL WITH BR. INSTRUCTION
                                        ;;*****************************************************************
                                        ;*TEST 2         POWER FAIL WITH BR. INSTRUCTION
                                        ;;*****************************************************************
                                        ;;
                                        TST2:
        002254
        002254  104401  002262                 TYPE    ,65$            ,;TYPE ASCIZ STRING
        002260  000402                          BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  *2 *
        002266
        002266  104401  003740                  TYPE,$ENULL
401     002272  005267  176622                 INC     $TESTN
402     002276  012706  001100                 MOV     #STACK,SP       ;SET UP STACK
403     002302  012767  000357  175466         MOV     #357,PS                 ;SET UP CONDITION CODES
404     002310  012703  002324                 MOV     #2$,R3                  ;SET POWER UP RETURN
405     002314  012777  000001  177074         MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
406     002322  000777                  1$:    BR      .                       ,WAIT FOR POWER FAIL
407     002324  010600                  2$:    MOV     SP,R0                   ;GET SP
408     002326  012706  001100                 MOV     #STACK,SP               ;SET STACK
409     002332  022700  001074                 CMP     #STACK-4,R0             ;CHECK STACK POINTER
410     002336  001414                         BEQ     3$                      ;SKIP IF OK
411     002340  012767  001074  005322         MOV     #1074,EXP.SP1
412     002346  010067  005320                 MOV     R0,REC.SP1
413     002352  012706  000500                 MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
        002356  104000                          ERROR

                                                                                ;ERROR
                                                                                ;-----
414                                                                             ;STACK POINTER
415     002360  007670                  EXP.SP1                                 ;PRINT RECEIVED SP
416     002362  007672                  REC.SP1                                 ;PRINT EXPECTED SP
417     002364  000000                  0
418     002366  000000                  HALT
419     002370  013700  001076          3$:    MOV     @#STACK-2,R0            ;GET OLD PS
420     002374  022700  000341                 CMP     #341,R0                 ;CHECK OLD PS
421     002400  001414                         BEQ     4$                      ;SKIP IF OK
422     002402  012767  000341  005264         MOV     #341,EXP.PSW
423     002410  010067  005262                 MOV     R0,REC.PSW
424     002414  012706  000500                 MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
        002420  104000                          ERROR

                                                                                ;ERROR
                                                                                ;-----
425                                                                             ;OLD PS IS WRONG
426     002422  007674                  EXP.PSW                                 ;PRINT EXPECTED OLD PS
427     002424  007676                  REC.PSW                                 ;PRINT RECEIVED OLD PS
428     002426  000000                  0
429     002430  000000                  HALT
430     002432  0 ,700  001074          4$:    MOV     @#STACK-4,R0            ;GET RETURN ADDRESS
431     002436  022700  002322                 CMP     #1$,R0
432     002442  001414                         BEQ     5$                      ;SKIP IF OK
433     002444  012767  002322  005212         MOV     #1$,EXP.ADD
434     002452  010067  005210                 MOV     R0,REC.ADD
435     002456  012706  000500                 MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
        002462  104000                          ERROR
```

```
                                                                        ;ERROR
                                                                        ;-----
436                                                                     ;ADDRESS ON STACK IS WRONG
437 002464  007664                      EXP.ADD                        ;PRINT EXPECTED ADDRESS
438 002466  007666                      REC.ADD                        ;PRINT RECEIVE ADDRESS
439 002470  000000                      0
440 002472  000000                      HALT
441 002474                          5$:
    002474  032737  040000  177570      BIT     #SW14,@#SWR            ;LOOP ON TEST?
    002502  001264                      BNE     TST2                   ;LOOP TO TST2
442
443                                 ;TEST ROUTINE TO CHECK RESTART CAPABILITY
444                                 ;USING THE EMULATOR TRAP FOR A WAIT
445                                 ;OPERATOR MUST SET HALT SWITCH TO ENABLE POSITION
446                                 ;
```

```
                                .SBTTL  TEST # 3 - POWER FAIL WITH EMT TRAP
                                ;;*****************************************************************
                                ;*TEST 3         POWER FAIL WITH EMT TRAP
                                ;;*****************************************************************
                                ;
                                TST3:
002504                                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
002504  104401  002512                  BR      64$                     ;;GET OVER THE ASCIZ
002510  000402                  ;;65$:  .ASCIZ  *3 *
                                64$:
002516                                  TYPE,$ENULL
002516  104401  003740
448 002522  005267  176372              INC     $TESTN
449 002526  005037  177776              CLR     @#PS                    ;SET KERNEL MODE
450 002532  012706  001100              MOV     #STACK,SP                       ;SET UP STACK
451 002536  012703  002570              MOV     #2$,R3                          ;SET POWER UP RETURN
452 002542  012767  001422  175260      MOV     #LRTI,EMTVEC                    ;SET UP RETURN FROM  EMT TRAP VECTOR
453 002550  012767  000005  175254      MOV     #5,EMTVEC+2                     ;SET PSW ON POWER FAIL
454 002556  012777  000001  176632      MOV     #1,@PFAIL                       ;SET UP TO START POWER FAIL
455 002564  104000              3$:     EMT                                             ;WAIT FOR POWER FAIL
456 002566  000776                      BR      3$
457 002570  010600              2$:     MOV     SP,R0                   ;GET SP
458 002572  012706  001100              MOV     #STACK,SP
459 002576  012767  006550  175224      MOV     #$ERROR,30              ;RESET EMT VECTOR
460 002604  022700  001074              CMP     #STACK-4,R0             ;WAS STACK PUSHED ONLY TWICE
461 002610  00146                       BEQ     6$
462 002612  022700  001070              CMP     #STACK-10,R0            ;WAS STACK PUSHED 4 TIMES
463 002616  001414                      BEQ     4$
464 002620  012767  001070  005042      MOV     #1070,EXP.SP1
465 002626  010067  005040              MOV     R0,REC.SP1
466 002632  012706  000500              MOV     #STACK1,SP                              ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
002636  104000                          ERROR

                                                                                ;ERROR
                                                                                ;-----
467                                                                             ;WAS STACK PUSHED 4 TIMES
468 002640  007670                      EXP.SP1                                 ;PRINT EXPECTED STACK POINTER
469 002642  007672                      REC.SP1                                 ;PRINT RECEIVED STACK POINTER
470 002644  000000                      0
471 002646  000000                      HALT
472 002650  013700  001070      4$:     MOV     @#STACK-10,R0
473 002654  022700  001422              CMP     #LRTI,R0        ;DOES STACK CONTAIN CORRECT INFO
474 002660  001414                      BEQ     5$
475 002662  012767  001422  004774      MOV     #LRTI,EXP.ADD
476 002670  010067  004772              MOV     R0,REC.ADD
477 002674  012706  000500              MOV     #STACK1,SP                              ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
002700  104000                          ERROR

                                                                                ;ERROR
                                                                                ;-----
478                                                             ;IS STACK PUSHED MORE THAN 2 OR LESS THAN 4
479 002702  007664                      EXP.ADD                                 ;PRINT EXPECTED ADDRESS
480 002704  007666                      REC.ADD                                 ;PRINT RECEIVED ADDRESS
481 002706  000000                      0
482 002710  000000                      HALT
483 002712  013700  001072      5$:     MOV     @#STACK-6,R0
484 002716  022700  000005              CMP     #5,R0           ;DOES STACK CONTAIN CORRECT INFO
```

```
                                                                                    K 2
    485 002722  001414                         BEQ     6$
    486 002724  012767  000005  004742         MOV     #5,EXP.PSW
    487 002732  000000  007676                 R0,REC.PSW
    488 002736  012706  000500                 MOV     #STACK1,SP                  ;SETUP RESERVED STACK POINTER
                                                                                    ;FOR ERROR PRINTOUT
        002742  104000                         ERROR

                                                                                ;ERROR
                                                                                ;-----
    489                                                                         ;STACK CONTIAN WRONG PSW
    490 002744  007674                         EXP.PSW                          ;PRINT EXPECTED PSW
    491 002746  007676                         REC.PSW                          ;PRINT RECEIVED PSW
    492 002750  000000                         0
    493 002752  000000                         HALT
    494 002754                          6$:
        002754  032737  040000  177570         BIT     #SW14,@#SWR     ;LOOP ON TEST?
        002762  001250                         BNE     TST3            ;LOOP TO TST3
```

```
    495                                         .SBTTL  TEST # 4 - POWER FAIL WITH ODD ADDRESS
                                                ;;*****************************************************************
                                                ;*TEST 4         POWER FAIL WITH ODD ADDRESS
                                                ;;*****************************************************************
         002764                                 TST4:
         002764  104401  002772                        TYPE    ,65$            ;;TYPE ASCIZ STRING
         002770  000402                                BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  *4 *
         002776                                 64$:
         002776  104401  003740                        TYPE,$ENUL
    496  003002  005267  176112                        INC     $TESTN
    497  003006  005037  177776                        CLR     @#PS            ;SET KERNEL MODE
    498  003012  012737  003032  000004                MOV     #3$,@#ERRVEC    ;SET TRAP VECTOR
    499  003020  012703  003052                        MOV     #1$,R3          ;SET RETURN ADDRESS FOR POWER FAIL
    500  003024  012777  000001  176364                MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
    501  003032  012706  001100                3$:     MOV     #STACK,SP       ;RESET STACK
    502  003036  005737  000003                        TST     @#3             ;CAUSE ODD ADDRESS TRAP
    503  003042  012706  000500                        MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                                ;FOR ERROR PRINTOUT
         003046  104000                                 ERROR

                                                                                        ;ERROR
                                                                                        ;-----
    504  003050  000000                                HALT                    ;ODD ADDRESS TRAP FAILED TO OCCUR
    505  003052  012737  000006  000004        1$:     MOV     #6,@#ERRVEC     ;RESET 4
    506  003060  032737  040000  177570                BIT     #SW14,@#SWR     ;LOOP ON TEST?
         003066  001336                                BNE     TST4            ;LOOP TO TST4
```

```
                                .SBTTL  TEST # 5 - POWER FAIL WITH TIME OUT (KERNEL)
                                ;;*****************************************************************
                                ;*TEST 5         POWER FAIL WITH TIME OUT (KERNEL)
                                ;;*****************************************************************
                                TST5:
 003070                                 TYPE    ,65$                    ;;TYPE ASCIZ STRING
 003070  104401  003076                 BR      64$                     ;;GET OVER THE ASCIZ
 003074  000402                 ;;65$:  .ASCIZ  *5 *
                                64$:
 003102                                 TYPE,$ENULL
 003102  104401  003740
508 003106  005267  176006             INC     $TESTN
509 003112  012737  003132  000004     MOV     #3$,@#ERRVEC            ;SET TRAP VECTOR
510 003120  012703  003156             MOV     #1$,R3                  ;SET UP RETURN ADDRESS FOR POWER FAIL
511 003124  012777  000001  176264     MOV     #1,@PFAIL                       ;SET UP TO START POWER FAIL
512 003132  012706  001100             MOV     #STACK,SP       3$:     ;SET STACK
513 003136  005037  177776             CLR     @#PS                    ;SET KERNEL MODE
514 003142  010037  160000             MOV     R0,@#160000             ;CAUSE A TIMEOUT
515 003146  012706  000500             MOV     #STACK1,SP                              ;SETUP RESERVED STACK POINTER
                                                                                       ;FOR ERROR PRINTOUT
 003152  104000                         ERROR

                                                                                ;ERROR
                                                                                ;-----
516 003154  000000                     HALT                            ;TIME OUT FAILED TO OCCUR
517 003156  012706  001100             MOV     #STACK,SP       1$:     ;SET STACK
518 003162  012737  000006  000004     MOV     #6,@#ERRVEC             ;RESET 4
519 003170  032737  040000  177570     BIT     #SW14,@#SWR             ;LOOP ON TEST?
 003176  001334                         BNE     TST5                    ;LOOP TO TST5
```

```
                                .SBTTL  TEST # 6 - POWER FAIL IN THE STACK OVERFLOW (KERNEL)
                                ;;*********************************************************************
                                ;*TEST 6         POWER FAIL IN THE STACK OVERFLOW (KERNEL)
                                ;;*********************************************************************
                                TST6:
003200                                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
003200  104401  003206                  BR      64$                     ;;GET OVER THE ASCIZ
003204  000402                  ;;65$:  .ASCIZ  *6 *
003212                          64$:
003212  104401  003740                  TYPE,$ENULL
521 003216  005267  175676              INC     $TESTN
522 003222  005037  177776              CLR     @#PS                    ;SET KERNEL MODE
523 003226  005067  001330              CLR     FLAG                    ;CLEAR THE FLAG
524 003232  012737  003272  000004      MOV     #2$,@#ERRVEC            ;SET SICK TRAP ADDRESS
525 003240  012706  000400              MOV     #400,SP                 ;SET STACK TO YELLOW ZONE
526 003244  012703  003266              MOV     #1$,R3                  ;SET RETURN ADDRESS FOR POWER FAIL
527 003250  012777  000001  176140      MOV     #1,@PFAIL                       ;SET UP TO START POWER FAIL
528 003256  000001                      WAIT                            ;WAIT FOR  THE POWER FAIL
529 003260  012706  000500              MOV     #STACK1,SP                              ;SETUP RESERVED STACK POINTER
                                                                                        ;FOR ERROR PRINTOUT
003264  104000                          ERROR

                                                                                ;ERROR
                                                                                ;-----
530 003266  000000              1$:     HALT
531 003270  000407                      BR      4$                      ;SKIP SP CHECK
532 003272  012737  000006  000004  2$:  MOV     #6,@#ERRVEC             ;RESET 4
533 003300  012703  003310          5$:  MOV     #4$,R3                  ;SET RETURN
534 003304  000002                      RTI                             ;GO TO THE POWER FAIL ROUTINE
535 003306  000000                      HALT
536 003310                      4$:
003310  032737  040000  177570          BIT     #SW14,@#SWR             ;LOOP ON TEST?
003316  001330                          BNE     TST6                    ;LOOP TO TST6
```

```
537                                     .SBTTL  TEST # 7 - POWER FAIL WITH RESETS
                                        ;;*****************************************************************
                                        ;*TEST 7         POWER FAIL WITH RESETS
                                        ;;*****************************************************************
                                        TST7:
003320
003320  104401  003326                        TYPE    ,65$                    ;;TYPE ASCIZ STRING
003324  000402                                BR      64$                     ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  *7 *
003332
003332  104401  003740                        TYPE,$ENULL
538 003336  005267  175556                    INC     $TESTN
539 003342  005037  177776                    CLR     @#PS                    ;SET KERNEL MODE
540 003346  012703  003404                    MOV     #1$,R3                  ;SET RETURN ADDRESS
541 003352  012706  001100                    MOV     #STACK,SP               ;RESET STACK
542 003356  012777  000001  176032            MOV     #1,@PFAIL                       ;SET UP TO START POWER FAIL
543 003364  000005                    3$:     RESET                           ;RESETS
544 003366  000005                            RESET                           ;TO WAIT
545 003370  000005                            RESET                           ;IN
546 003372  000774                            BR      3$                      ;LOOP
547 003374  012706  000500                    MOV     #STACK1,SP                              ;SETUP RESERVED STACK POINTER
                                                                                            ;FOR ERROR PRINTOUT
003400  104000                                ERROR

                                                                                    ;ERROR
                                                                                    ;-----
548                                                                         ;RESET LOOP FAILED
549 003402  000000                            HALT
550 003404  012706  001100            1$:     MOV     #STACK,SP               ;RESET STACK
551 003410  032737  040000  177570            BIT     #SW14,@#SWR             ;LOOP ON TEST?
003416  001340                                BNE     TST7                    ;LOOP TO TST7
```

```
552                                     .SBTTL  TEST # 10 - MEMORY MANAGEMENT ABORT TEST
                                        ;;*****************************************************************
                                        ;*TEST 10        MEMORY MANAGEMENT ABORT TEST
                                        ;;*****************************************************************
                                        TST10:
    003420
    003420  104401  003426                      TYPE    ,65$                    ;;TYPE ASCIZ STRING
    003424  000402                              BR      64$                     ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  *10 *
    003432
    003432  104401  003740              64$:    TYPE,$ENULL
553 003436  005267  175456                      INC     $TESTN
554 003442  005037  177776                      CLR     @#PS                    ;SET KERNEL MODE
555 003446  012737  003522  000004              MOV     #4$,@#ERRVEC            ;SET FOR TIMEOUT
556 003454  004767  001110                      JSR     PC,MAP  ;MAP THE WORLD
557 003460  012737  003504  000250              MOV     #3$,@#MMVEC             ;SET MEMORY MANAGEMENT VECTOR
558 003466  012703  003524                      MOV     #1$,R3                  ;LOAD PF RETURN
559 003472  012777  000001  175716              MOV     #1,@PFAIL                       ;SET UP TO START POWER FAIL
560 003500  005237  177572                      INC     @#MMR0                  ;TURN MEMORY MANAGEMENT ON
561 003504  012706  001100          3$:         MOV     #STACK,SP               ;ZAP STACK
562 003510  005237  140000                      INC     @#140000                ;ACCESS VIOLATION
563 003514  012706  000500                      MOV     #STACK1,SP                              ;SETUP RESERVED STACK POINTER
                                                                                                ;FOR ERROR PRINTOUT
    003520  104000                              ERROR

                                                                                        ;ERROR
                                                                                        ;-----
564                                                                             ;NO VIOLATION OR TRAP TO 4
565 003522  000000                  4$:         HALT
566
567 003524  005037  177572          1$:         CLR     @#MMR0                  ;TURN OFF MEMORY MANAGEMENT
568 003530  012706  001100          2$:         MOV     #STACK,SP               ;MAKE A NEW STACK
569 003534  012737  000006  000004              MOV     #6,@#ERRVEC             ;RESET 4
570 003542  032737  040000  177570              BIT     #SW14,@#SWR             ;LOOP ON TEST?
    003550  001323                              BNE     TST10                   ;LOOP TO TST10
```

```
571                                             .SBTTL  TEST # 11 - MEMORY VOLATILITY TEST
                                                ;;*****************************************************************
                                                ;*TEST 11        MEMORY VOLATILITY TEST
                                                ;;*****************************************************************
    003552  000240                              TST11:  NOP
572 003554  005267  175340                              INC     $TESTN
573 003560  005037  177776                              CLR     @#PS            ;SET KERNEL MODE
574 003564  004767  000170                      4$:     JSR     PC,LOAD         ;LOAD ALL MEMORY WITH 52525
575 003570  012703  003626                              MOV     #1$,R3          ;POWER FAIL RETURN ADDRESS
576 003574  104401  003602                              TYPE    ,65$            ;;TYPE ASCIZ STRING
    003600  000402                              BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  *11 *
    003606                                      64$:
    003606  104401  003740                              TYPE,$ENULL
577 003612  004767  000246                      2$:     JSR     PC,CHECK        ;CHECK FOR THE 52525
578 003616  012777  000001  175572                      MOV     #1,@PFAIL               ;SET UP TO START POWER FAIL
579 003624  000772                              BR      2$              ,LOOP FOR EVER OR POWER FAIL
580 003626  012706  001100                      1$:     MOV     #STACK,SP       ;ZAP THE STACK
581 003632  004767  000226                              JSR     PC,CHECK        ;CHECK ALL MEMORY
582 003636  032737  040000  177570                      BIT     #SW14,@#SWR     ;LOOP ON TEST?
    003644  001342                              BNE     TST11           ;LOOP TO TST11
```

```
                        .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 PRETST
003646                  $EOP:
003646  000240                NOP
003650  005267  175246        INC     $PASS           ;;INCREMENT THE PASS NUMBER
003654  042767  100000  175240  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
003662  005327                DEC     (PC)+           ;;LOOP?
003664  000001          $EOPCT: .WORD   1
003666  003022                BGT     $DOAGN          ;;YES
003670  012737                MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
003672  000001          $ENDCT: .WORD   1
003674  003664                $EOPCT
003676  104401  003743        TYPE    ,$ENDMG         ;;TYPE 'END PASS #''
003702  016746  175214        MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
003706  104405                TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
003710  104401  003740        TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
003714  013700  000042  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
003720  001405                BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
003722  000005                RESET                   ;;CLEAR THE WORLD
003724  004710          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
003726  000240                NOP                     ;;SAVE ROOM
003730  000240                NOP                     ;;FOR
003732  000240                NOP                     ;;ACT11
003734                  $DOAGN:
003734  000137                JMP     @(PC)+          ;;RETURN
003736  002020          $RTNAD: .WORD   PRETST
003740     377     377  000  $ENULL: .BYTE   -1,-1,0       ;;NULL CHARACTER STRING
003743     015     012  105  $ENDMG: .ASCIZ  <15><12>/END PASS #/
```

```
586                                             .SBTTL  LOAD MEMORY WITH DATA PATTERN
587 003760  004767  000604          LOAD:       JSR     PC,MAP          ;SET UP MEMORY MANAGEMENT REGISTERS
588 003764  016704  000574                      MOV     LIMIT,R4        ;GET BANK COUNT
589 003770  016705  000572                      MOV     DATA,R5         ;GET THE DATA PATTERN
590 003774  012737  004024  000250              MOV     #2$,@#MMVEC     ;SET UP FOR MEMORY MANAGEMENT ABORTS
591 004002  012700  010502                      MOV     #END,R0         ;FORM ADDRESS OF WHERE TO START WRITING THE DATA
592 004006  062700  120000                      ADD     #120000,R0      ;USE KIPAR5 FOR ADDRESSING
593 004012  012737  000001  177572              MOV     #1,@#MMR0       ;TURN ON MEMORY MANAGEMENT
594 004020  010520                  1$:         MOV     R5,(R0)+        ;WRITE THE DATA
595 004022  000776                              BR      1$
596 004024  005304                  2$:         DEC     R4              ;LAST BANK WRITTEN?
597 004026  100411                              BMI     3$              ;BRANCH IF YES
598 004030  012700  120000                      MOV     #120000,R0      ;RESET ADDRESSING REGISTER
599 004034  062737  000200  172352              ADD     #200,@#KIPAR5   ;SIZE NEXT 4K BANK
600 004042  012737  000001  177572              MOV     #1,@#MMR0       ;CLEAR MEMORY MANAGEMENT ERRORS
601 004050  000002                              RTI                     ;RETURN TO CONTINUE WRITING
602 004052  005037  177572          3$:         CLR     @#MMR0          ;TURN OFF MEMORY MANAGEMENT
603 004056  062706  000004                      ADD     #4,SP           ;CLEAN UP THE STACK
604 004062  000207                              RTS     PC              ;RETURN TO CALLER
```

```
                                        .SBTTL  CHECK MEMORY FOR CORRECT DATA PATTERN

004064  004767  000500         CHECK:  JSR     PC,MAP          ;SET UP MEMORY MANAGEMENT REGISTERS
004070  016704  000470                 MOV     LIMIT,R4        ;GET BANK COUNT
004074  016705  000466                 MOV     DATA,R5         ;GET DATA PATTERN
004100  012737  004220  000250         MOV     #2$,@#MMVEC     ;SET UP FOR MEMORY MANAGEMENT ABORTS
004106  012700  010502                 MOV     #END,R0         ;ADDRESS OF WHERE DATA PATTERN STARTS
004112  062700  120000                 ADD     #120000,R0      ;USE KIPAR5 FOR ADDRESSING
004116  012737  000001  177572         MOV     #1,@#MMR0       ;TURN ON MEM. MANAGEMENT
004124  012001                 1$:     MOV     (R0)+,R1        ;READ THE DATA
004126  020501                 CMP     R5,R1           ;IS IT GOOD?
004130  001775                 BEQ     1$              ;BRANCH IF YES
004132  010067  003542                 MOV     R0,VIR.ADD
004136  162767  000002  003534         SUB     #2,VIR.ADD
004144  042767  160000  003526         BIC     #160000,VIR.ADD
004152  013767  172352  003522         MOV     @#KIPAR5,PAR.OFF
004160  010567  003474                 MOV     R5,EXPDAT
004164  010167  003472                 MOV     R1,RECDAT
004170  012706  000500                 MOV     #STACK1,SP                      ;SETUP RESERVED STACK POINTER
                                                                                ;FOR ERROR PRINTOUT
004174  104000                 ERROR

                                                                        ;ERROR
                                                                        ;-----
004176  007700                 VIR.ADD                 ;PRINT VIRTUAL ADDRESS
004200  007702                 PAR.OFF                 ;PRINT PDR OFFSET
004202  007660                 EXPDAT                  ;PRINT EXPECTED DATA
004204  007662                 RECDAT                  ;PRINT RECEIVED DATA
004206  000000                 0
004210  000000                 HALT
004212  010560  177776                 MOV     R5,-2(R0)       ;WRITE GOOD DATA
004216  000742                 BR      1$              ;GO READ THE NEXT WORD
004220  005304         2$:     DEC     R4              ;LAST BANK COMPARED?
004222  100411                 BMI     3$              ;BRANCH IF YES
004224  012700  120000                 MOV     #120000,R0      ;RESET ADDRESSING REGISTER
004230  062737  000200  172352         ADD     #200,@#KIPAR5   ;SIZE NEXT 4K BANK
004236  012737  000001  177572         MOV     #1,@#MMR0       ;CLEAR MEMORY MANAGEMENT ERRORS
004244  000002                 RTI                     ;CONTINUE TESTING
004246  005037  177572         3$:     CLR     @#MMR0          ;TURN OFF MEM. MANAGEMENT
004252  062706  000004                 ADD     #4,SP           ;CLEAN UP THE STACK
004256  000207                 RTS     PC              ;RETURN FOR CALL
```

```
                                                                    .SBTTL  PARITY ERROR HANDLER
                                                            PARERR: D'      (PC)+           ;FIRST TIME IN?
                                                            PARFLG:         1
                                                                    .       2$              ;BRANCH IF YES
                                                            1$:     .                       ;NO--ANOTHER PARITY ERROR OCCURRED WHILE
                                                                    BR      1$              ;PROCESSING THE FIRST ONE.
004260  005327                                              2$:     BIT     #77,HIADRS      ;MSB'S OF PARITY ERROR ADDRESS=0?
004262  000001                                                      BNE     4$              ;BRANCH IF NO
004264  002002                                                      CMP     @#LOADRS,#END   ;LSB'S ABOVE THE PROGRAM?
004266  000000                                                      BHI     4$              ;BRANCH IF YES
004270  000776                                              3$:     HALT                    ;PARITY ERROR IS IN THE PROGRAM
004272  032767  000077  173442                                    BR      3$
004300  001006                                              4$:     HALT                    ;DATA PARITY ERROR OCCURRED
004302  023727  177740  010502                                    TST     @#MEMERR        ;DID AN ABORT OCCUR?
004310  101002                                                      BMI     5$              ;BRANCH IF YES
004312  000000                                                      SUB     #2,R0           ;BACKUP BY ONE WORD
004314  000776                                                      MOV     #BIT01,R5
004316  000000                                                      XOR     R5,R0           ;
004320  005737  177744                                    5$:     MOV     R3,(R0)+        ;REWRITE THE BAD DATA
004324  100405                                                      MOV     @#MEMERR,@#MEMERR ;CLEAR ERROR INDICATORS
004326  162700  000002                                            MOV     #1,PARFLG       ;INITIALIZE PARITY ERROR FLAG
004332  012705  000002                                            RTI                     ;CONTINUE TESTING
004336  074500
004340  010320
004342  013737  177744  177744
004350  012767  000001  177704
004356  000002
```

POWER FAIL ROUTINE

```
                                        .SBTTL  POWER FAIL ROUTINE
                                        ;;*****************************************************************
004360  012767  177777  000174  POWDWN: MOV     #-1,FLAG        ;FIRST INSTRUCTION FLAG
004366  005067  000170                  CLR     FLAG            ;NOW CLEAR IT
004372  012777  004522  000150          MOV     #ILLUP,@UVEC    ;IF TOO FAST
004400  011667  000142                  MOV     (SP),ERRAD      ;SET THE ERROR ADDRESS
004404  022706  000400                  CMP     #400,SP         ;YELLOW OR RED?
004410  100402                          BMI     1$              ;NO
004412  012706  001100                  MOV     #STACK,SP       ;SET EMERGENCY STACK
004416  010046                  1$:     MOV     R0,-(R6)        ;PUT
004420  010146                          MOV     R1,-(R6)        ;THE
004422  010246                          MOV     R2,-(R6)        ;REGISTERS
004424  010346                          MOV     R3,-(R6)        ;ON
004426  010446                          MOV     R4,-(R6)        ;THE
004430  010546                          MOV     R5,-(R6)        ;STACK
004432  010667  000122                  MOV     SP,SAV6         ;SAVE THE STACK POINTER
004436  004737  172240                  JSR     PC,@#SIPAR0     ;SET TIME FACTOR
004442  012777  004454  000100          MOV     #POWUP,@UVEC    ;RESET THE UP VECTOR
004450  000000                          HALT                    ;WAIT FOR POWER DOWN
004452  000240                          NOP
                                        ;
004454  016706  000100          POWUP:  MOV     SAV6,SP         ;RESET SP
004460  012777  004534  000066          MOV     #ILLDWN,@DVEC   ;SET TOO FAST DOWN VECTOR
004466  004767  000200                  JSR     PC,SETPAR       ;GO PUT TIME FACTOR SUBROUTINE IN SIPAR0-SIPAR3
004472  004737  172240                  JSR     PC,@#SIPAR0     ;SET TIME FACTOR
004476  012605                          MOV     (R6)+,R5        ;TAKE
004500  012604                          MOV     (R6)+,R4        ;THE
004502  012603                          MOV     (R6)+,R3        ;REGISTERS
004504  012602                          MOV     (R6)+,R2        ;FROM
004506  012601                          MOV     (R6)+,R1        ;THE
004510  012600                          MOV     (R6)+,R0        ;STACK
004512  012777  004360  000034          MOV     #POWDWN,@DVEC   ;RESET THE DOWN VECTOR
004520  000113                          JMP     (R3)            ;JUMP INDIRECT TO R3
                                        ;
004522  104401                  ILLUP:  TYPE                    ;POWER UP BEFORE POWER DOWN COMPLETE
004524  010037                          UPF
004526  000000                          HALT
004530  000167  173444                  JMP     200
004534  104401                  ILLDWN: TYPE                    ;POWER DOWN BEFORE UP COMPLETE
004536  010107                          DOWN
004540  000000                          HALT
004542  000167  173432                  JMP     200
                                        ;
                                        ;
004546  000000                  ERRAD:  .WORD   0               ;RETURN ADDRESS FROM POWER FAIL
004550  000024  000026          UVEC:   .WORD   24,26           ;UP ADDRESS PAIR
004554  000024  000026          DVEC:   .WORD   24,26           ;DOWN ADDRESS PAIR
004560  000000                  SAV6:   .WORD   0               ;SOME PLACE TO PUT THE SP
004562  000000                  FLAG:   .WORD   0               ;1 INSTRUCTION DOWN FLAG
004564  000000                  LIMIT:  .WORD   0               ;TOP OF MEMORY
004566  052525                  DATA:   .WORD   52525           ;WHAT IS TO BE WRITTEN INTO MEMORY
```

```
                                        .SBTTL  SETUP MEMORY MANAGMENT REGISTERS
                                
004570  012737  000000  172340  MAP:    MOV     #0,@#KIPAR0     ;SETUP PAR0 FOR 1ST 4K
004576  012737  077406  172300          MOV     #77406,@#KIPDR0 ;4K, R/W, EXPAND UP
004604  012737  000000  172352          MOV     #0,@#KIPAR5     ;SET UP PAR5 FOR 1ST 4K
004612  012737  077406  172312          MOV     #77406,@#KIPDR5 ;4K, R/W, ED=UP
004620  012737  000200  172354          MOV     #200,@#KIPAR6   ;SET UP PAR6 FOR 2ND 4K
004626  012737  000000  172314          MOV     #0,@#KIPDR6     ;ABORT ALL REFERENCES
004634  012737  177600  172356          MOV     #177600,@#KIPAR7;SET UP PAR7 FOR I/O PAGE
004642  012737  077406  172316          MOV     #77406,@#KIPDR7 ;4K, R/W, ED=UP
004650  012737  000020  172516          MOV     #BIT04,@#MMR3   ;SET UP FOR 22-BIT MAPPING
004656  012737  004666  000250          MOV     #MMERR,@#MMVEC  ;SET UP MEMORY MANAGEMENT VECTOR
004664  000207                          RTS     PC              ;RETURN FROM CALL
004666  000000                  MMERR:  HALT                    ;MEMORY MANAGEMENT ERROR
004670  000776                          BR      MMERR
```

```
                                        .SBTTL  SUBROUTINE TO LOAD SIPAR0-SIPAR3 WITH A SUBROUTINE
004672  012737  012700  172240  SETPAR: MOV     #12700,@#SIPAR0 ;LOAD PAR0 & 1 WITH MOV #1777,R0
004700  012737  001777  172242          MOV     #1777,@#SIPAR1
004706  012737  077001  172244          MOV     #77001,@#SIPAR2 ;LOAD PAR2 WITH SOB R0,.
004714  012737  000207  172246          MOV     #207,@#SIPAR3   ;LOAD PAR3 WITH RTS PC
004722  000207                          RTS     PC
                                .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
004724  010046                  $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
004726  010146                          MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
004730  010246                          MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
004732  010346                          MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
004734  010446                          MOV     R4,-(SP)        ;;SAVE R4 ON THE STACK                  BK001
004736  013746  000114                  MOV     @#114,-(SP)     ;;SAVE MEMORY ERROR VECTOR PS & PC
004742  013746  000116                  MOV     @#116,-(SP)
004746  012737  000116  000114          MOV     #116,@#114      ;;IGNORE PARITY ERRORS WHILE SIZING
004754  012737  000002  000116          MOV     #RTI,@#116
004762  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
004766  013746  000006                  MOV     @#ERRVEC+2,-(SP)
004772  010600                          MOV     SP,R0           ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
004774  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
004776  012637  000006                  MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
005002  012701  003776                  MOV     #3776,R1        ;;SETUP ADDRESS
005006  105727                          TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
005010  000200                  $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
005012  100145                          BPL     $CORE           ;;BR IF NO
005014  012737  005320  000004          MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
005022  005737  177572                  TST     @#SR0           ;;KT11 ARE YOU THERE?
005026  052767  100000  177754          BIS     #100000,$KT11   ;;YES--SET KT11 KEY
005034  012737  005064  000004          MOV     #100$,@#ERRVEC  ;;SET FOR TIMEOUT                       BK001
005042  005737  170200                  TST     @#170200        ;;UNIBUS MAP ARE YOU THERE?             BK001
005046  012737  176200  005104          MOV     #176200,@#$STOP ;;YES-SET COMPARISON VALUE FOR 11/70    BK001
005054  012737  000200  005102          MOV     #200,@#$MAP     ;;TURN ON MAP INDICATOR                 BK001
005062  000411                          BR      $MAPRG          ;;GO SET UP MAP REGISTERS               BK001
005064  012737  006200  005104  100$:   MOV     #6200,@#$STOP   ;;COMPARISON VALUE FOR 18 BIT MAPPING   BK001
005072  022626                          CMP     (SP)+,(SP)+     ;;CLEAN OFF STACK                       BK001
005074  005037  005102                  CLR     @#$MAP          ;;MAKE SURE MAP INDICATOR TURNED OFF    BK001
005100  000412                          BR      $NOMAP          ;;                              BK001
005102  000000                  $MAP:   .WORD   0               ;;=200 IF MAP PRESENT                   BK001
005104  000000                  $STOP:  .WORD   0               ;;FILLED WITH APPROPRIATE COMPARISON VALUE      BK001
005106  012703  000037          $MAPRG: MOV     #37,R3          ;;SET UP COUNTER                        BK001
005112  012702  170200                  MOV     #170200,R2      ;;START WITH MAPL0                      BK001
005116  005022                  100$:   CLR     (R2)+           ;;LOAD ALL MAP REGISTERS                BK001
```

```
005120  012722  000074                  MOV     #74,(R2)+       ;;WITH THE VALUE 17000000                BK001
005124  077304                          SOB     R3,100$         ;;DO ALL 31 REGISTERS                    BK001
005126                          $NOMAP:
005126  005046                          CLR     -(SP)           ;;INITIALIZE FOR 'PAR' LOADING
005130  012702  172340                  MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST 'PAR'
005134  012703  000010                  MOV     #^D8,R3         ;;LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
005140  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
005146  011622                          MOV     (SP),(R2)+      ;;LOAD 'PAR'
005150  062716  000200                  ADD     #200,(SP)       ;;UPDATE FOR NEXT 'PAR'
005154  077307                          SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
005156  012742  177600                  MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
005162  005042                          CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
005164  012737  005202  000004          MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
005172  012737  000060  172516          MOV     #60,@#SR3       ;;ENABLE 22 BIT MODE AND UNIBUS MAP      BK001
005200  000401                          BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
005202  022626                  2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
005204  005237  177572          3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
005210  012737  005256  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
005216  105737  005102                  TSTB    @#$MAP          ;;IS MAP THERE?                          BK001
005222  100006                          BPL     4$              ;;NO-SKIP                                BK001
005224  012737  005300  000114          MOV     #$MMOUT,@#114   ;;SET UP MEMORY ERROR VECTOR             BK001
005232  013737  000006  000116          MOV     @#ERRVEC+2,@#116 ;;LOCK OUT INTERRUPTS           BK001
005240  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
005244  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
005250  023712  005104                  CMP     @#$STOP,(R2)    ;;LAST ONE?
005254  101371                          BHI     4$              ;;NO--TRY IT
005256  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
005260  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
005264  105737  005102                  TSTB    @#$MAP          ;;IS MAP THERE?                          BK001
005270  100034                          BPL     $SIZEX          ;;NO-SKIP                                BK001
005272  005037  172516                  CLR     @#SR3           ;;TURN OFF MAP                           BK001
005276  000431                          BR      $SIZEX
005300  013704  177744          $MMOUT: MOV     @#177744,R4     ;;SAVE MEMORY ERROR REGISTER             BK001
005304  010437  177744                  MOV     R4,@#177744     ;;CLEAR BITS IN REGISTER                 BK001
005310  032704  000001                  BIT     #1,R4           ;;MEMORY TIMEOUT?                        BK001
005314  001360                          BNE     $KTOUT          ;;YES-EXIT                               BK001
005316  000002                          RTI                     ;;MUST BE PARITY ERROR-IGNORE IT         BK001
005320  042767  100000  177462  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
005326  012737  005356  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
005334  005002                          CLR     R2              ;;SET UP BANK
005336  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
005342  062702  000040                  ADD     #40,R2          ;;1K STEP
005346  005711                          TST     (R1)            ;;TRAP ON TIME OUT
005350  022701  177776                  CMP     #177776,R1      ;;LAST ONE
005354  001370                          BNE     1$              ;;NO--TRY AGAIN
005356  162701  004000          $CROUT: SUB     #4000,R1
005362  162702  000040          $SIZEX: SUB     #40,R2          ;;DROP BACK
005366  010006                          MOV     R0,SP           ;;RESTORE THE STACK
005370  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
005374  012637  000004                  MOV     (SP)+,@#ERRVEC
005400  012637  000116                  MOV     (SP)+,@#116     ;;RESTORE MEMORY ERROR VECTOR
005404  012637  000114                  MOV     (SP)+,@#114
005410  010167  000020                  MOV     R1,$LSTAD       ;;LAST ADDRESS
005414  010267  000016                  MOV     R2,$LSTBK       ;;LAST BANK
005420  012604                          MOV     (SP)+,R4        ;;RESTORE R4                             BK001
005422  012603                          MOV     (SP)+,R3        ;;RESTORE R3
005424  012602                          MOV     (SP)+,R2        ;;RESTORE R2
```

```
005426  012601                  MOV     (SP)+,R1        ;;RESTORE R1
005430  012600                  MOV     (SP)+,R0        ;;RESTORE R0
005432  000207                  RTS     PC
005434  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
005436  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
```

```
739                                     .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
005440  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
005444  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
005452  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
005456  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
005462  000406                          BR      $TYPON
005464  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
005472  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
005500  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
005506  010346                          MOV     R3,-(SP)        ;;SAVE R3
005510  010446                          MOV     R4,-(SP)        ;;SAVE R4
005512  010546                          MOV     R5,-(SP)        ;;SAVE R5
005514  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
005520  005404                          NEG     R4
005522  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
005526  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
005532  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
005536  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
005542  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
005544  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
005546  000404                          BR      3$              ;;GO DO MSB
005550  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
005552  006105                          ROL     R5
005554  006105                          ROL     R5
005556  010503                          MOV     R5,R3
005560  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
005562  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
005566  100016                          BPL     7$              ;;BR IF NO
005570  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
005574  001002                          BNE     4$              ;;TEST FOR 0
005576  005704                          TST     R4              ;;SUPPRESS THIS 0?
005600  001403                          BEQ     5$              ;;BR IF YES
005602  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
005604  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
005610  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
```

BINARY TO OCTAL (ASCII) AND TYPE

```
005614  110367  000040            MOVB    R3,8$           ;;SAVE FOR TYPING
005620  104401  005660            TYPE    ,8$             ;;GO TYPE THIS DIGIT
005624  105367  000032    7$:     DECB    $OCNT           ;;COUNT BY 1
005630  003347                    BGT     2$              ;;BR IF MORE TO DO
005632  002402                    BLT     6$              ;;BR IF DONE
005634  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
005636  000744                    BR      2$              ;;GO DO THE LAST DIGIT
005640  012605            6$:     MOV     (SP)+,R5        ;;RESTORE R5
005642  012604                    MOV     (SP)+,R4        ;;RESTORE R4
005644  012603                    MOV     (SP)+,R3        ;;RESTORE R3
005646  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
005654  012616                    MOV     (SP)+,(SP)
005656  000002                    RTI                     ;;RETURN
005660     000            8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
005661     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
005662     000            $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
005663     000            $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
005664  000000            $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
741                                     .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
                                        $TYPDS:
005666
005666  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
005670  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
005672  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
005674  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
005676  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
005700  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
005704  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
005710  100004                          BPL     1$              ;;BR IF INPUT IS POS.
005712  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
005714  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
005722  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
005724  012703  006102                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
005730  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
005734  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
005736  016001  006072                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
005742  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
005744  002402                          BLT     4$              ;;BR IF DONE
005746  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
005750  000774                          BR      3$
005752  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
005754  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
005756  001002                          BNE     5$              ;;FALL THROUGH IF 0
005760  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
005762  100407                          BMI     7$              ;;BR IF YES
005764  106316                  5$:     ASLB    (SP)            ;;MSD?
005766  103003                          BCC     6$              ;;BR IF NO
005770  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
005776  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
006002  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
006006  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
006010  005720                          TST     (R0)+           ;;JUST INCREMENTING
006012  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
006016  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
006020  003002                          BGT     8$              ;;GO TO EXIT
006022  010502                          MOV     R5,R2           ;;GET THE LSD
006024  000764                          BR      6$              ;;GO CHANGE TO ASCII
006026  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
006030  100003                          BPL     9$              ;;BR IF NO
006032  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
006040  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
006042  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
006044  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
006046  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
006050  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
006052  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
006054  104401  006102                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
```

```
006060  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
006066  012616                          MOV     (SP)+,(SP)
006070  000002                          RTI                     ;;RETURN TO USER
006072  023420                  $DTBL:  10000.
006074  001750                          1000.
006076  000144                          100.
006100  000012                          10.
006102                          $DBLK:  .BLKW   4
```

```
                        .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
                        ;*
006112  105767  000353  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
006116  100002                 BPL     1$              ;;BR IF YES
006120  000000                 HALT                    ;;HALT HERE IF NO TERMINAL
006122  000430                 BR      3$              ;;LEAVE
006124  010046          1$:    MOV     R0,-(SP)        ;;SAVE R0
006126  017600  000002         MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
006132  122767  000001  172774 CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
006140  001011                 BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
006142  132767  000100  172765 BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
006150  001405                 BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
006152  010067  000004         MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
006156  004767  172774         JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
006162  000000          61$:   .WORD   0               ;;MESSAGE ADDRESS
006164  132767  000040  172743 62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
006172  001003                 BNE     60$             ;;YES,SKIP TYPE OUT
006174  112046          2$:    MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
006176  001005                 BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
006200  005726                 TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
006202  012600          60$:   MOV     (SP)+,R0        ;;RESTORE R0
006204  062716  000002  3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
006210  000002                 RTI                     ;;RETURN
006212  122716  000011  4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
006216  001430                 BEQ     8$
006220  122716  000200         CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
006224  001006                 BNE     5$
006226  005726                 TST     (SP)+           ;;POP  <CR><LF> EQUIV
006230  104401                 TYPE                    ;;TYPE A CR AND LF
006232  006473                 $CRLF
006234  105067  000212         CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
006240  000755                 BR      2$              ;;GET NEXT CHARACTER
006242  004767  000056  5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
006246  126726  000216  6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
006252  001350                 BNE     2$              ;;IF NO GO GET NEXT CHAR.
006254  016746  000206         MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                       ;;AND THE NULL CHAR.
006260  105366  000001  7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
006264  002770                 BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
006266  004767  000032         JSR     PC,$TYPEC       ;;GO TYPE A NULL
006272  105367  000154         DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
006276  000770                 BR      7$              ;;LOOP
                        ;HORIZONTAL TAB PROCESSOR
006300  112716  000040  8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
```

```
006304  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
006310  132767  000007  000134          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
006316  001372                          BNE     9$              ;;TAB STOP
006320  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
006322  000724                          BR      2$              ;;GET NEXT CHARACTER
006324                          $TYPEC:
006324  105777  000126                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
006330  100022                          BPL     10$             ;;BR IF NOT                     ;MJD001
006332  017746  000122                  MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
006336  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
006342  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
006346  001012                          BNE     102$            ;;BR IF NOT                     ;MJD001
006350                          101$:                                                           ;MJD001
006350  105777  000102                  TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
006354  100375                          BPL     101$                                            ;MJD001
006356  117716  000076                  MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
006362  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
006366  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
006372  001366                          BNE     101$            ;;BR IF NOT                     ;MJD001
006374                          102$:                                                           ;MJD001
006374  005726                          TST     (SP)+           ;;FIX STACK                     ;MJD001
006376                          10$:                                                            ;MJD001
006376  105777  000060                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
006402  100375                          BPL     10$                                             ;MJD001
006404  126627  000002  000021          CMPB    2(SP),#$XON     ;;IS CHARACTER A RANDOM XON?    ;RAN001
006412  001420                          BEQ     $TYPEX          ;;BRANCH IF YES                 ;RAN001
006414  116677  000002  000042          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
006422  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
006430  001003                          BNE     1$              ;;BRANCH IF NO
006432  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
006436  000406                          BR      $TYPEX          ;;EXIT
006440  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
006446  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
006450  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
006452  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
006454  000207                  $TYPEX: RTS     PC
006456  177560                  $TKS:   .WORD   177560          ;;TTY KDB STATUS                ;MJD001
006460  177562                  $TKB:   .WORD   177562          ;;TTY KBD BUFFER                ;MJD001
006462  177564                  $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
006464  177566                  $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
006466     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
006467     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
006470     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
006471     000                  $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07> 0 YES)
006472     077                  $QUES:  .ASCII  "?"             ;;QUESTION MARK
006473     015                  $CRLF:  .ASCII  <15>            ;;CARRAIGE RETURN
006474     012     000          $LF:    .ASCIZ  <12>            ;;LINEFEED
```

```
745                                     .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.
006476  010046                          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
006500  016600  000002                          MOV     2(SP),R0        ;;GET TRAP ADDRESS
006504  005740                          .       TST     -(R0)           ;;BACKUP BY 2
006506  111000                          .       MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
006510  006300                          .       ASL     R0              ;;POSITION FOR INDEXING
006512  016000  006532                  .       MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
006516  000200                          .       RTS     R0              ;;GO TO ROUTINE
                                        ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
006520  011646                          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
006522  016666  000004  000002          .       MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
006530  000002                          .       RTI                     ;;RESTORE THE PSW
                                        .SBTTL  TRAP TABLE
                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE ''TRAP'' INSTRUCTION.
                                        ;       ROUTINE
                                        ;       -------
006532  006520                          $TRPAD: .WORD   $TRAP2
006534  006112                          .       $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
006536  005464                          .       $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
006540  005440                          .       $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
006542  005500                          .       $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
006544  005666                          .       $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
```

```
                                        .SBTTL  ERROR PRINT ROUTINE
                        104000          ERROR   =104000
006546  000000          ERRPC:  .WORD   0
006550  010067  000026  $ERROR: MOV     R0,SAVR0        ;SAVE R0 THRU R5
006554  010167  000024          MOV     R1,SAVR1
006560  010267  000022          MOV     R2,SAVR2
006564  010367  000020          MOV     R3,SAVR3
006570  010467  000016          MOV     R4,SAVR4
006574  010567  000014          MOV     R5,SAVR5
006600  000406                  BR      TITL
006602  000000          SAVR0:  .WORD
006604  000000          SAVR1:  .WORD
006606  000000          SAVR2:  .WORD
006610  000000          SAVR3:  .WORD
006612  000000          SAVR4:  .WORD
006614  000000          SAVR5:  .WORD

006616  011601          TITL:   MOV     (SP),R1         ;R1 CONTAINS ADDRESS FLLOWING ERRPC ADDRESS
006620  162701  000002          SUB     #2,R1
006624  010167  177716          MOV     R1,ERRPC        ;GET ADDRESS OF ERROR MESSAGE:ERRPC
006630  104401  006636          TYPE    ,65$            ;;TYPE ASCIZ STRING
006634  000411                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <CRLF><CRLF>/TESTNO    ERRPC/
006660                  64$:
006660  005721          3$:     TST     (R1)+           ;R1 POINTS TO NEXT ARGUEMENT

006662  012702  007634          MOV     #PRTABL,R2      ;ADDRESS OF START OF PRINT TABLE LIST
006666  012703  007110          MOV     #PRTITL,R3      ;ADDRESS OF STERT OF TITLES
006672  005711                  TST     (R1)            ;END OF ARGUEMENTS?
006674  001405                  BEQ     DAT             ;YES,GO PRINT DATA
006676  005723          2$:     TST     (R3)+           ;INDEX THRU TITLES
006700  021122                  CMP     (R1),(R2)+      ;SEARCH PRINT TABLE LIST FOR TITLE
006702  001375                  BNE     2$              ;NO; CHECK NEXT LOCATION IN LIST

006704  004753                  JSR     PC,@-(R3)       ;FOUND IT; GO PRINT TITLE CODE ERROR ROUTINE
006706  000764                  BR      3$

006710                  DAT:
006710  104401  006716          TYPE    ,65$            ;;TYPE ASCIZ STRING
006714  000401                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <CRLF>
006720                  64$:
006720  016746  172174          MOV     $TESTN,-(SP)    ;;SAVE $TESTN FOR TYPEOUT
006724  104403                  TYPOS                   ;;GO TYPE--OCTAL ASCII
006726     006                  .BYTE   6               ;;TYPE 6 DIGIT(S)
006727     000                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
006730  104401  006736          TYPE    ,67$            ;;TYPE ASCIZ STRING
006734  000402                  BR      66$             ;;GET OVER THE ASCIZ
                        ;;67$:  .ASCIZ  /  /
006742                  66$:
006742  016746  177600          MOV     ERRPC,-(SP)     ;;SAVE ERRPC FOR TYPEOUT
006746  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
006750  104401  003740          TYPE    ,$ENULL

006754  011601                  MOV     (SP),R1         ;R1 CONTAINS ADDRESS FOLLOWING ERRORPC ADDRESS
006756  162701  000002          SUB     #2,R1
006762  005721          3$:     TST     (R1)+           ;R1 POINTS TO NEXT ARGUEMENT
```

```
                                        MOV     #PRTABL,R2      ;ADDRESS OF START OF PRINT TABLE LIST
                                        MOV     #PRDATA,R3      ;ADDRESS OF STERT OF PRINT DATA
                                        TST     (R1)            ;END OF ARGUEMENTS?
                                        BEQ     FIN             ;YES
                                2$:     TST     (R3)+           ;INDEX THRU DATA PRINTS
                                        CMP     (R1),(R2)+      ;SEARCH PRINT TABLE LIST FOR TITLE
                                        BNE     2$              ;NO; CHECK NEXT LOCATION IN LIST
                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  /   /
                                68$:
                                        JSR     PC,@-(R3)       ;
                                        TYPE    ,$ENULL
                                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                                        BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  /   /
                                70$:
                                        BR      3$

                                FIN:    TST     (R1)+           ;R1 NOW CONTAINS RETURN ADDRESS
                                        MOV     R1,(SP)         ;SETUP RETURN ADDRESS IN STACK
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>
                                64$:

                                        MOV     SAVR0,R0        ;RESTORE REGISTERS
                                        MOV     SAVR1,R1
                                        MOV     SAVR2,R2
                                        MOV     SAVR3,R3
                                        MOV     SAVR4,R4
                                        MOV     SAVR5,R5

                                RTI                             ;RETURN
```

```
                                                .SBTTL  SUBROUTINES TO PRINT DATA HEADERS
007110  007134  007162  007210  PRTITL: .WORD   1$,2$,3$,4$,5$,6$,7$,10$,11$,12$
007134                          1$:
007134  104401  007142                 TYPE    ,65$            ;;TYPE ASCIZ STRING
007140  000407                         BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /   EXPDAT   /
007160                          64$:
007160  000207                         RTS PC
007162                          2$:
007162  104401  007170                 TYPE    ,67$            ;;TYPE ASCIZ STRING
007166  000407                         BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /   RECDAT   /
007206                          66$:
007206  000207                         RTS PC
007210                          3$:
007210  104401  007216                 TYPE    ,69$            ;;TYPE ASCIZ STRING
007214  000407                         BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  /   EXP.ADD   /
007234                          68$:
007234  000207                         RTS PC
007236                          4$:
007236  104401  007244                 TYPE    ,71$            ;;TYPE ASCIZ STRING
007242  000407                         BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  /   REC.ADD   /
007262                          70$:
007262  000207                         RTS PC
007264                          5$:
007264  104401  007272                 TYPE    ,73$            ;;TYPE ASCIZ STRING
007270  000407                         BR      72$             ;;GET OVER THE ASCIZ
                                ;;73$:  .ASCIZ  /   EXP.SP1   /
007310                          72$:
007310  000207                         RTS PC
007312                          6$:
007312  104401  007320                 TYPE    ,75$            ;;TYPE ASCIZ STRING
007316  000407                         BR      74$             ;;GET OVER THE ASCIZ
                                ;;75$:  .ASCIZ  /   REC.SP1   /
007336                          74$:
007336  000207                         RTS PC
007340                          7$:
007340  104401  007346                 TYPE    ,77$            ;;TYPE ASCIZ STRING
007344  000407                         BR      76$             ;;GET OVER THE ASCIZ
                                ;;77$:  .ASCIZ  /   EXP.PSW   /
007364                          76$:
007364  000207                         RTS PC
007366                          10$:
007366  104401  007374                 TYPE    ,79$            ;;TYPE ASCIZ STRING
007372  000407                         BR      78$             ;;GET OVER THE ASCIZ
                                ;;79$:  .ASCIZ  /   REC.PSW   /
007412                          78$:
007412  000207                         RTS PC
007414                          11$:
007414  104401  007422                 TYPE    ,81$            ;;TYPE ASCIZ STRING
007420  000407                         BR      80$             ;;GET OVER THE ASCIZ
                                ;;81$:  .ASCIZ  /   VIR.ADD   /
007440                          80$:
007440  000207                         RTS PC
007442                          12$:
```

```
        007442  104401  007450                  TYPE    ,83$            ;;TYPE ASCIZ STRING
        007446  000407                          BR      82$             ;;GET OVER THE ASCIZ
                                        ;;83$:  .ASCIZ  /   PAR.OFF   /
        007466                          82$:
    836 007466  000207                          RTS PC
```

```
                                                .SBTTL  SUBROUTINES TO PRINT DATA
007470  007514  007524  007534  PRDATA: .WORD   1$,2$,3$,4$,5$,6$,7$,10$,11$,12$
007514                          1$:
007514  016746  000140                 MOV     EXPDAT,-(SP)    ;;SAVE EXPDAT FOR TYPEOUT
007520  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007522  000207                         RTS PC
007524                          2$:
007524  016746  000132                 MOV     RECDAT,-(SP)    ;;SAVE RECDAT FOR TYPEOUT
007530  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007532  000207                         RTS PC
007534                          3$:
007534  016746  000124                 MOV     EXP.ADD,-(SP)   ;;SAVE EXP.ADD FOR TYPEOUT
007540  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007542  000207                         RTS PC
007544                          4$:
007544  016746  000116                 MOV     REC.ADD,-(SP    ;;SAVE REC.ADD FOR TYPEOUT
007550  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007552  000207                         RTS PC
007554                          5$:
007554  016746  000110                 MOV     EXP.SP1,-(SP)   ;;SAVE EXP.SP1 FOR TYPEOUT
007560  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007562  000207                         RTS PC
007564                          6$:
007564  016746  000102                 MOV     REC.SP1,-(SP)   ;;SAVE REC.SP1 FOR TYPEOUT
007570  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007572  000207                         RTS PC
007574                          7$:
007574  016746  000074                 MOV     EXP.PSW,-(SP)   ;;SAVE EXP.PSW FOR TYPEOUT
007600  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007602  000207                         RTS PC
007604                          10$:
007604  016746  000066                 MOV     REC.PSW,-(SP)   ;;SAVE REC.PSW FOR TYPEOUT
007610  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007612  000207                         RTS PC
007614                          11$:
007614  016746  000060                 MOV     VIR.ADD,-(SP)   ;;SAVE VIR.ADD FOR TYPEOUT
007620  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007622  000207                         RTS PC
007624                          12$:
007624  016746  000052                 MOV     PAR.OFF,-(SP)   ;;SAVE PAR.OFF FOR TYPEOUT
007630  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
007632  000207                         RTS PC
```

```
                                                .SBTTL  TABLES AND ASCII MESSAGES
007634  007660  007662  007664  PRTABL: .WORD   EXPDAT,RECDAT,EXP.ADD,REC.ADD,EXP.SP1
007646  007672  007674  007676          .WORD   REC.SP1,EXP.PSW,REC.PSW,VIR.ADD,PAR.OFF
007660  000000                  EXPDAT: .WORD   0
007662  000000                  RECDAT: .WORD   0
007664  000000                  EXP.ADD: .WORD  0
007666  000000                  REC.ADD: .WORD  0
007670  000000                  EXP.SP1: .WORD  0
007672  000000                  REC.SP1: .WORD  0
007674  000000                  EXP.PSW: .WORD  0
007676  000000                  REC.PSW: .WORD  0
007700  000000                  VIR.ADD: .WORD  0
007702  000000                  PAR.OFF: .WORD  0
007704     015     012     103  PNAME:  .ASCIZ <15><12>+CKKACC0 11/44 POWER FAIL+
007737     015     012     103  TITLE:  .ASCIZ <15><12>+CKKACC0 11/44 POWER FAIL+<15><12>
007774     015     012     102  BOOT:   .ASCIZ <15><12>+BOOT ENABLE SWITCH MUST BE OFF+<15><12>
010037     015     012     120  UPF:    .ASCIZ <15><12>+POWER UP BEFORE POWER DOWN COMPLETE+<15><12>
010107     015     012     120  DOWN:   .ASCIZ <15><12>+POWER DOWN BEFORE UP COMPLETE+<15><12>
010151     055     113     040  SUPMSG: .ASCII  ?-K WORDS OF MEMORY EXIST?<CRLF>
010202     122     101     116          .ASCII  ?RANDOM CHARACTER(S) (INCLUDING FORM FEED) ?
010254     115     101     131          .ASCIZ  ?MAY PRINT ON POWER CYCLING.?<CRLF>
010311     115     101     116  USRPFL: .ASCIZ  ?MANUALLY INTERRUPT THE POWER AFTER EACH TEST #?<CRLF>
010371     116     117     040  HDWPFL: .ASCII  ?NO MANUAL INTERVENTION NECESSARY - ?
010434     101     125     124          .ASCIZ  ?AUTO POWER FAIL TEST HARDWARE EXISTS?<CRLF>
                                        .EVEN
010502  000000                  END:    0
        000001                  .END
```

SYMBOL TABLE

```
ABASE = 000000        BIT03 = 000010        KDPAR0= 172360        MAPH14= 170262        MAPL33= 170354
ACDW1 = 000000        BIT04 = 000020        KDPAR1= 172362        MAPH15= 170266        MAPL34= 170360
ACDW2 = 000000        BIT05 = 000040        KDPAR2= 172364        MAPH16= 170272        MAPL35= 170364
ACPUOP= 000000        BIT06 = 000100        KDPAR3= 172366        MAPH17= 170276        MAPL36= 170370
ADDW0 = 000000        BIT07 = 000200        KDPAR4= 172370        MAPH2 = 170212        MAPL37= 170374
ADDW1 = 000000        BIT08 = 000400        KDPAR5= 172372        MAPH20= 170302        MAPL4 = 170220
ADDW10= 000000        BIT09 = 001000        KDPAR6= 172374        MAPH21= 170306        MAPL5 = 170224
ADDW11= 000000        BIT1  = 000002        KDPAR7= 172376        MAPH22= 170312        MAPL6 = 170230
ADDW12= 000000        BIT10 = 002000        KDPDR0= 172320        MAPH23= 170316        MAPL7 = 170234
ADDW13= 000000        BIT11 = 004000        KDPDR1= 172322        MAPH24= 170320        MEMERR= 177744
ADDW14= 000000        BIT12 = 010000        KDPDR2= 172324        MAPH25= 170326        MMERR   004666
ADDW15= 000000        BIT13 = 020000        KDPDR3= 172326        MAPH26= 170332        MMR0  = 177572
ADDW2 = 000000        BIT14 = 040000        KDPDR4= 172330        MAPH27= 170336        MMR1  = 177574
ADDW3 = 000000        BIT15 = 100000        KDPDR5= 172332        MAPH3 = 170216        MMR2  = 177576
ADDW4 = 000000        BIT2  = 000004        KDPDR6= 172334        MAPH30= 170342        MMR3  = 172516
ADDW5 = 000000        BIT3  = 000010        KDPDR7= 172336        MAPH31= 170346        MMVEC = 000250
ADDW6 = 000000        BIT4  = 000020        KERSTK= 001100        MAPH32= 170352        PARERR  004260
ADDW7 = 000000        BIT5  = 000040        KIPAR0= 172340        MAPH33= 170356        PARFLG  004262
ADDW8 = 000000        BIT6  = 000100        KIPAR1= 172342        MAPH34= 170362        PAR.OF  007702
ADDW9 = 000000        BIT7  = 000200        KIPAR2= 172344        MAPH35= 170366        PFAIL   001416
ADEVCT= 000000        BIT8  = 000400        KIPAR3= 172346        MAPH36= 170372        PIRQ  = 177772
ADEVM = 000000        BIT9  = 001000        KIPAR4= 172350        MAPH37= 170376        PIRQVE= 000240
AENV  = 000000        BOOT    007774        KIPAR5= 172352        MAPH4 = 170222        PNAME   007704
AENVM = 000000        BPTVEC= 000014        KIPAR6= 172354        MAPH5 = 170226        POWDWN  004360
AFATAL= 000000        CACHVE= 000114        KIPAR7= 172356        MAPH6 = 170232        POWUP   004454
AMADR1= 000000        CHECK   004064        KIPDR0= 172300        MAPH7 = 170236        PRDATA  007470
AMADR2= 000000        CONTRL  177746        KIPDR1= 172302        MAPL0 = 170200        PRETST  002020
AMADR3= 000000        CPUERR= 177766        KIPDR2= 172304        MAPL00= 170200        PRTABL  007634
AMADR4= 000000        CR    = 000015        KIPDR3= 172306        MAPL01= 170204        PRTITL  007110
AMAMS1= 000000        CRLF  = 000200        KIPDR4= 172310        MAPL02= 170210        PR0   = 000000
AMAMS2= 000000        DAT     006710        KIPDR5= 172312        MAPL03= 170214        PR1   = 000040
AMAMS3= 000000        DATA    004566        KIPDR6= 172314        MAPL04= 170220        PR2   = 000100
AMAMS4= 000000        DDISP = 177570        KIPDR7= 172316        MAPL05= 170224        PR3   = 000140
AMSGAD= 000000        DISPLA= 177570        KSP   =%000006        MAPL06= 170230        PR4   = 000200
AMSGLG= 000000        DISPRE  000174        LF    = 000012        MAPL07= 170234        PR5   = 000240
AMSGTY= 000000        DOWN    010107        LIMIT   004564        MAPL1 = 170204        PR6   = 000300
AMTYP1= 000000        DSWR  = 177570        LKS   = 177546        MAPL10= 170240        PR7   = 000340
AMTYP2= 000000        DVEC    004554        LKVEC = 000100        MAPL11= 170244        PS    = 177776
AMTYP3= 000000        EMTVEC= 000030        LOAD    003760        MAPL12= 170250        PSW   = 177776
AMTYP4= 000000        END     010502        LOADRS= 177740        MAPL13= 170254        PWRVEC= 000024
APASS = 000000        ERRAD   004546        LRTI    001422        MAPL14= 170260        RECDAT  007662
APRIOR= 000000        ERROR = 104000        MAINT = 177750        MAPL15= 170264        REC.AD  007666
APTCSU= 000040        ERRPC   006546        MAP     004570        MAPL16= 170270        REC.PS  007676
APTENV= 000001        ERRVEC= 000004        MAPH0 = 170202        MAPL17= 170274        REC.SP  007672
APTSIZ= 000200        EXPDAT  007660        MAPH00= 170202        MAPL2 = 170210        RESVEC= 000010
APTSPO= 000100        EXP.AD  007664        MAPH01= 170206        MAPL20= 170300        R10   =%000000
ASWREG= 000000        EXP.PS  007674        MAPH02= 170212        MAPL21= 170304        R11   =%000001
ATESTN= 000000        EXP.SP  007670        MAPH03= 170216        MAPL22= 170310        R12   =%000002
AUNIT = 000000        FIN     007042        MAPH04= 170222        MAPL23= 170314        R13   =%000003
AUSWR = 000000        FLAG    004562        MAPH05= 170226        MAPL24= 170320        R14   =%000004
AVECT1= 000000        HDWPFL  010371        MAPH06= 170232        MAPL25= 170324        R15   =%000005
AVECT2= 000000        HIADRS= 177742        MAPH07= 170236        MAPL26= 170330        R6    =%000006
BEGIN   001424        HITMIS= 177752        MAPH1 = 170206        MAPL27= 170334        R7    =%000007
BIT0  = 000001        HT    = 000011        MAPH10= 170242        MAPL3 = 170214        SAVR0   006602
BIT00 = 000001        ILLDWN  004534        MAPH11= 170246        MAPL30= 170340        SAVR1   006604
BIT01 = 000002        ILLUP   004522        MAPH12= 170252        MAPL31= 170344        SAVR2   006606
BIT02 = 000004        IOTVEC= 000020        MAPH13  170256        MAPL32= 170350        SAVR3   006610
```

SYMBOL TABLE

```
SAVR4   006612
SAVR5   006614
SAV6    004560
SCOPE = 000004
SDPAR0= 172260
SDPAR1= 172262
SDPAR2= 172264
SDPAR3= 172266
SDPAR4= 172270
SDPAR5= 172272
SDPAR6= 172274
SDPAR7= 172276
SDPDR0= 172220
SDPDR1= 172222
SDPDR2= 172224
SDPDR3= 172226
SDPDR4= 172230
SDPDR5= 172232
SDPDR6= 172234
SDPDR7= 172236
SETPAR  004672
SIPAR0= 172240
SIPAR1= 172242
SIPAR2= 172244
SIPAR3= 172246
SIPAR4= 172250
SIPAR5= 172252
SIPAR6= 172254
SIPAR7= 172256
SIPDR0= 172200
SIPDR1= 172202
SIPDR2= 172204
SIPDR3= 172206
SIPDR4= 172210
SIPDR5= 172212
SIPDR6= 172214
SIPDR7= 172216
SIZEHI= 177762
SIZELO= 177760
SR0   = 177572
SR1   = 177574
SR2   = 177576
SR3   = 172516
SSP   =%000006
STACK = 001100
STACK1= 000500
STKLMT= 177774
SUPMSG  010151
SUPSTK= 000700
SWR   = 177570
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7     000200
SW8   = 000400
SW9   = 001000
SYSTID= 177764
TBITVE= 000014
TITL    006616
TITLE   007737
TKVEC = 000060
TPVEC = 000064
TRAPVE  000034
TRTVEC= 000014
TST1    002036
TST10   003420
TST11   003552
TST2    002254
TST3    002504
TST4    002764
TST5    003070
TST6    003200
TST7    003320
TYPDS = 104405
TYPE  = 104401
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
UDPAR0= 177660
UDPAR1= 177662
UDPAR2= 177664
UDPAR3= 177666
UDPAR4= 177670
UDPAR5= 177672
UDPAR6= 177674
UDPAR7= 177676
UDPDR0= 177620
UDPDR1= 177622
UDPDR2= 177624
UDPDR3= 177626
UDPDR4= 177630
UDPDR5= 177632
UDPDR6= 177634
UDPDR7= 177636
UIPAR0= 177640
UIPAR1= 177642
UIPAR2= 177644
UIPAR3= 177646
UIPAR4= 177650
UIPAR5= 177652
UIPAR6= 177654
UIPAR7= 177656
UIPDR0= 177600
UIPDR1= 177602
UIPDR2= 177604
UIPDR3= 177606
UIPDR4= 177610
UIPDR5= 177612
UIPDR6= 177614
UIPDR7= 177616
UPF     010037
USESTK= 000600
USP   =%000006
USRPFL  010311
UVEC    004550
VIR.AD  007700
XYZ     001420
$APTHD  001100
$ATYC   001174
$ATY1   001150
$ATY3   001156
$ATY4   001166
$CHARC  006452
$CORE   005326
$CPUOP  001142
$CRLF   006473
$CROUT  005356
$DBLK   006102
$DEVCT  001124
$DOAGN  003734
$DTBL   006072
$ENDAD  003724
$ENDCT  003672
$ENDMG  003743
$ENULL  003740
$ENV    001134
$ENVM   001135
$EOP    003646
$EOPCT  003664
$ERROR  006550
$ETABL  001134
$ETEND  001150
$FATAL  001116
$FFLG   001414
$FILLC  006470
$FILLS  006467
$GET42  003714
$HD   = 000000
$HIBTS  001100
$KTNEX  005320
$KTOUT  005256
$KT11   005010
$LF     006474
$LFLG   001413
$LSTAD  005434
$LSTBK  005436
$MADR1  001146
$MAIL   001114
$MAMS1  001144
$MAP    005102
$MAPRG  005106
$MBADR  001102
$MFLG   001412
$MMOUT  005300
$MSGAD  001130
$MSGLG  001132
$MSGTY  001114
$MTYP1  001145
$NOMAP  005126
$NULL   006466
$NWTST= 000001
$OCNT   005662
$OMODE  005664
$PASS   001122
$PASTM  001106
$QUES   006472
$RTNAD  003736
$SAVPC= 000204
$SETUP= 000024
$SIZE   004724
$SIZEX  005362
$STOP   005104
$STUP = 177777
$SVPC = 000204
$SWR  = 040000
$SWREG  001136
$TESTN  001120
$TKB    006460
$TKS    006456
$TN   = 000012
$TPB    006464
$TPFLG  006471
$TPS    006462
$TRAP   006476
$TRAP2  006520
$TRP  = 000006
$TRPAD  006532
$TSTM   001104
$TYPDS  005666
$TYPE   006112
$TYPEC  006324
$TYPEX  006454
$TYPOC  005464
$TYPON  005500
$TYPOS  005440
$UNIT   001126
$UNITM  001110
$USWR   001140
$XOFF = 000023
$XON  = 000021
$$GET4= 000000
$OFILL  005663
.$X   = 001100

. ABS.  010504     000
        000000     001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  33040 WORDS  ( 130 PAGES)
DYNAMIC MEMORY:  20034 WORDS  ( 77 PAGES)
ELAPSED TIME:  00:05:53
CKKACC.BIN,CKKACC/CR/-SP/NL:TOC=CKKACC.MLB/ML,CKKACC.P11
```

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES		
ABASE	- 000000	7-292		
ACDW1	= 000000	7-292		
ACDW2	= 000000	7-292		
ACPUOP	= 000000	7-292	7-292	
ADDW0	= 000000	7-292		
ADDW1	= 000000	7-292		
ADDW10	- 000000	7-292		
ADDW11	- 000000	7-292		
ADDW12	= 000000	7-292		
ADDW13	= 000000	7-292		
ADDW14	- 000000	7-292		
ADDW15	- 000000	7-292		
ADDW2	- 000000	7-292		
ADDW3	- 000000	7-292		
ADDW4	000000	7-292		
ADDW5	- 000000	7-292		
ADDW6	= 000000	7-292		
ADDW7	= 000000	7-292		
ADDW8	= 000000	7-292		
ADDW9	= 000000	7-292		
ADEVCT	= 000000	7-292	7-292	
ADEVM	= 000000	7-292		
AENV	= 000000	7-292	7-292	
AFNVM	= 000000	7-292	7-292	
AFATAL	= 000000	7-292	7-292	
AMADR1	= 000000	7-292	7-292	
AMADR2	= 000000	7-292		
AMADR3	= 000000	7-292		
AMADR4	= 000000	7-292		
AMAMS1	= 000000	7-292	7-292	
AMAMS2	= 000000	7-292		
AMAMS3	= 000000	7-292		
AMAMS4	= 000000	7-292		
AMSGAD	= 000000	7-292	7-292	
AMSGLG	= 000000	7-292	7-292	
AMSGTY	= 000000	7-292	7-292	
AMTYP1	- 000000	7-292	7-292	
AMTYP2	= 000000	7-292		
AMTYP3	= 000000	7-292		
AMTYP4	- 000000	7-292		
APASS	= 000000	7-292	7-292	
APRIOR	= 000000	7-292		
APTCSU	000040	#7-293	28-743	
APTENV	= 000001	7-293	#7-293	28-743
APTSIZ	= 000200	#7-293	8-300	
APTSPO	- 000100	7-293	#7-293	28-743
ASWREG	- 000000	7-292	7-292	
ATESTN	= 000000	7-292	7-292	
AUNIT	- 000000	7-292	7-292	
AUSWR	= 000000	7-292	7-292	
AVECT1	- 000000	7-292		
AVECT2	- 000000	7-292		

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
BEGIN	001424	7-281	#8-300							
BIT0	= 000001	#7-280								
BIT00	= 000001	#7-280	7-280							
BIT01	= 000002	#7-280	7-280	22-658						
BIT02	= 000004	#7-280	7-280							
BIT03	= 000010	#7-280	7-280							
BIT04	= 000020	#7-280	7-280	24-725						
BIT05	= 000040	#7-280	7-280							
BIT06	= 000100	#7-280	7-280							
BIT07	= 000200	#7-280	7-280	8-309						
BIT08	= 000400	#7-280	7-280							
BIT09	= 001000	#7-280	7-280							
BIT1	= 000002	#7-280								
BIT10	= 002000	#7-280								
BIT11	= 004000	#7-280								
BIT12	= 010000	#7-280								
BIT13	= 020000	#7-280								
BIT14	= 040000	#7-280								
BIT15	= 100000	#7-280								
BIT2	= 000004	#7-280								
BIT3	= 000010	#7-280								
BIT4	= 000020	#7-280								
BIT5	= 000040	#7-280								
BIT6	= 000100	#7-280								
BIT7	= 000200	#7-280								
BIT8	= 000400	#7-280								
BIT9	= 001000	#7-280								
BOOT	007774	9-340	#33-874							
BPTVEC	= 000014	#7-280								
CACHVE	= 000114	#7-280	*8-302							
CHECK	004064	18-577	18-581	#21-607						
CONTRL	= 177746	#7-280								
CPUERR	= 177766	#7-280								
CR	= 000015	#7-280	28-743	28-743						
CRLF	= 000200	#7-280	28-743	28-743	30-767	30-767	30-781	30-805	33-877	33-879
		33-880	33-882							
DAT	006710	30-773	#30-781							
DATA	004566	20-589	21-609	#23-714						
DDISP	= 177570	#7-280	8-300							
DISPLA	= 177570	#7-298	*8-300	*8-300						
DISPRE	000174	#7-281	8-300							
DOWN	010107	23-703	#33-876							
DSWR	= 177570	#7-280	8-300							
DVEC	004554	8-301	23-686	23-695	#23-710					
EMTVEC	= 000030	#7-280	*12-452	*12-453						
END	010502	20-591	21-611	22-650	#33-884					
ERRAD	004546	*23-669	#23-708							
ERROR	= 104000	#7-280	10-365	10-377	10-388	11-413	11-424	11-435	12-466	12-477
		12-488	13-503	14-515	15-529	16-547	17-563	21-623	#30-748	
ERRPC	006546	#30-749	*30-766	30-784						
ERRVEC	000004	#7-280	8-300	*8-300	*8-300	*13-498	*13-505	*14-509	*14-518	*15-524
		*15-532	*17-555	*17-569	25-737	25-737	*25-737	*25-737	*25-737	*25-737

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
		*25-737	25-737	*25-737	*25-737	*25-737				
EXPDAT	C07660	*21-621	21-626	32-839	33-860	#33-862				
EXP.AD	007664	*10-375	10-379	*11-433	11-437	*12-475	12-479	32-843	33-860	#33-864
EXP.PS	007674	*10-386	10-390	*11-422	11-426	*12-486	12-490	32-851	33-861	#33-868
EXP.SP	007670	*10-363	10-367	*11-411	11-415	*12-464	12-468	32-847	33-860	#33-866
FIN	007042	30-793	#30-803							
FLAG	004562	*15-523	*23-666	*23-667	#23-712					
GNS	= ******	7-281	7-281	10-355	11-400	12-447	13-495	14-507	15-520	16-537
		17-552	18-576	29-745	29-745	29-745	29-745	29-745	29-745	29-745
		29-745	29-745	29-745	30-767	30-781	30-783	30-797	30-800	30-805
		31-817	31-819	31-821	31-823	31-825	31-827	31-829	31-831	31-833
		31-835								
HDWPFL	010371	9-345	#33-881							
HIADRS	= 177742	#7-280	22-648							
HITMIS	= 177752	#7-280								
HT	= 000011	#7-280	28-743	28-743						
ILLDWN	004534	23-686	#23-702							
ILLUP	004522	23-668	#23-698							
IOTVEC	= 000020	#7-280								
KDPAR0	= 172360	#7-280								
KDPAR1	= 172362	#7-280								
KDPAR2	= 172364	#7-280								
KDPAR3	= 172366	#7-280								
KDPAR4	= 172370	#7-280								
KDPAR5	= 172372	#7-280								
KDPAR6	= 172374	#7-280								
KDPAR7	= 172376	#7-280								
KDPDR0	= 172320	#7-280								
KDPDR1	= 172322	#7-280								
KDPDR2	= 172324	#7-280								
KDPDR3	= 172326	#7-280								
KDPDR4	= 172330	#7-280								
KDPDR5	= 172332	#7-280								
KDPDR6	= 172334	#7-280								
KDPDR7	= 172336	#7-280								
KERSTK	= 001100	#7-280								
KIPAR0	= 172340	#7-280	*24-717	25-737	25-737					
KIPAR1	= 172342	#7-280								
KIPAR2	= 172344	#7-280								
KIPAR3	= 172346	#7-280								
KIPAR4	= 172350	#7-280								
KIPAR5	= 172352	#7-280	*20-599	21-620	*21-635	*24-719				
KIPAR6	= 172354	#7-280	*24-721							
KIPAR7	= 172356	#7-280	*24-723							
KIPDR0	= 172300	#7-280	*24-718							
KIPDR1	= 172302	#7-280								
KIPDR2	= 172304	#7-280								
KIPDR3	= 172306	#7-280								
KIPDR4	= 172310	#7-280								
KIPDR5	= 172312	#7-280	*24-720							
KIPDR6	= 172314	#7-280	*24-722							
KIPDR7	= 172316	#7-280	*24-724							

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES			
KSP	=%000006	#7-280			
LF	= 000012	#7-280	28-743	28-743	
LIMIT	004564	*8-316	20-588	21-608	#23-713
LKS	= 177546	#7-280			
LKVEC	- 000100	#7-280			
LOAD	003760	18-574	#20-587		
LOADRS	= 177740	#7-280	22-650		
LRTI	001422	#7-299	12-452	12-473	12-475
MAINT	- 177750	#7-280			
MAP	004570	17-556	20-587	21-607	#24-717
MAPH0	= 170202	#7-280			
MAPH00	= 170202	#7-280	7-280		
MAPH01	- 170206	#7-280	7-280		
MAPH02	= 170212	#7-280	7-280		
MAPH03	- 170216	#7-280	7-280		
MAPH04	= 170222	#7-280	7-280		
MAPH05	= 170226	#7-280	7-280		
MAPH06	= 170232	#7-280	7-280		
MAPH07	- 170236	#7-280	7-280		
MAPH1	= 170206	#7-280			
MAPH10	- 170242	#7-280			
MAPH11	= 170246	#7-280			
MAPH12	- 170252	#7-280			
MAPH13	- 170256	#7-280			
MAPH14	= 170262	#7-280			
MAPH15	- 170266	#7-280			
MAPH16	- 170272	#7-280			
MAPH17	170276	#7-280			
MAPH2	- 170212	#7-280			
MAPH20	- 170302	#7-280			
MAPH21	= 170306	#7-280			
MAPH22	- 170312	#7-280			
MAPH23	= 170316	#7-280			
MAPH24	= 170320	#7-280			
MAPH25	- 170326	#7-280			
MAPH26	= 170332	#7-280			
MAPH27	= 170336	#7-280			
MAPH3	= 170216	#7-280			
MAPH30	= 170342	#7-280			
MAPH31	170346	#7-280			
MAPH32	- 170352	#7-280			
MAPH33	= 170356	#7-280			
MAPH34	= 170362	#7-280			
MAPH35	= 170366	#7-280			
MAPH36	= 170372	#7-280			
MAPH37	= 170376	#7-280			
MAPH4	= 170222	#7-280			
MAPH5	= 170226	#7-280			
MAPH6	= 170232	#7-280			
MAPH7	= 170236	#7-280			
MAPL0	- 170200	#7-280			
MAPL00	- 170200	#7-280	7-280		

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
MAPL01	= 170204	#7-280	7-280							
MAPL02	= 170210	#7-280	7-280							
MAPL03	= 170214	#7-280	7-280							
MAPL04	= 170220	#7-280	7-280							
MAPL05	= 170224	#7-280	7-280							
MAPL06	= 170230	#7-280	7-280							
MAPL07	= 170234	#7-280	7-280							
MAPL1	= 170204	#7-280								
MAPL10	- 170240	#7-280								
MAPL11	- 170244	#7-280								
MAPL12	= 170250	#7-280								
MAPL13	= 170254	#7-280								
MAPL14	= 170260	#7-280								
MAPL15	= 170264	#7-280								
MAPL16	= 170270	#7-280								
MAPL17	= 170274	#7-280								
MAPL2	- 170210	#7-280								
MAPL20	= 170300	#7-280								
MAPL21	170304	#7-280								
MAPL22	- 170310	#7-280								
MAPL23	- 170314	#7-280								
MAPL24	- 170320	#7-280								
MAPL25	= 170324	#7-280								
MAPL26	- 170330	#7-280								
MAPL27	- 170334	#7-280								
MAPL3	= 170214	#7-280								
MAPL30	- 170340	#7-280								
MAPL31	= 170344	#7-280								
MAPL32	- 170350	#7-280								
MAPL33	- 170354	#7-280								
MAPL34	170360	#7-280								
MAPL35	= 170364	#7-280								
MAPL36	= 170370	#7-280								
MAPL37	= 170374	#7-280								
MAPL4	= 170220	#7-280								
MAPL5	= 170224	#7-280								
MAPL6	= 170230	#7-280								
MAPL7	= 170234	#7-280								
MEMERR	- 177744	#7-280	22-655	22-661	*22-661					
MMERR	004666	24-726	#24-728	24-729						
MMR0	= 177572	#7-280	7-280	*17-560	*17-567	*20-593	*20-600	*20-602	*21-613	*21-636
		*21-638								
MMR1	177574	#7-280	7-280							
MMR2	= 177576	#7-280	7-280							
MMR3	- 172516	#7-280	7-280	*24-725						
MMVEC	000250	#7-280	*17-557	*20-590	*21-610	*24-726				
PARERR	004260	8-302	#22-643							
PARFLG	004262	*8-303	#22-644	*22-662						
PAR.OF	007702	*21-620	21-625	32-857	33-861	#33-871				
PFAIL	001416	#7-295	*8-323	8-326	*8-332	9-343	10-358	11-405	12-454	13-500
		14-511	15-527	16-542	17-559	18-578				
PIRQ	= 177772	#7-280								

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES							
PIRQVE	= 000240	#7-280							
PNAME	007704	#33-872							
POWDWN	004360	8-301	#23-666	23-695					
POWUP	004454	23-681	#23-685						
PRDATA	007470	30-791	#32-838						
PRETST	002020	9-338	9-346	#9-348	19-584				
PRTABL	007634	30-770	30-790	#33-860					
PRTITL	007110	30-771	#31-816						
PR0	= 000000	#7-280							
PR1	= 000040	#7-280							
PR2	= 000100	#7-280							
PR3	= 000140	#7-280							
PR4	= 000200	#7-280							
PR5	= 000240	#7-280							
PR6	= 000300	#7-280							
PR7	= 000340	#7-280							
PS	- 177776	#7-280	7-280	*10-356	*11-403	*12-449	*13-497	*14-513	*15-522	*16-539
		*17-554	*18-573						
PSW	= 177776	#7-280							
PWRVEC	= 000024	#7-280							
RECDAT	007662	*21-622	21-627	32-841	33-860	#33-863			
REC.AD	007666	*10-376	10-380	*11-434	11-438	*12-476	12-480	32-845	33-860	#33-865
REC.PS	007676	*10-387	10-391	*11-423	11-427	12-487	12-491	32-853	33-861	#33-869
REC.SP	007672	*10-364	10-368	*11-412	11-416	*12-465	12-469	32-849	33-861	#33-867
RESVEC	= 000010	#7-280	.							
R10	=%000000	#7-280							
R11	=%000001	#7-280							
R12	=%000002	#7-280							
R13	=%000003	#7-280							
R14	-%000004	#7-280							
R15	-%000005	#7-280							
R6	=%000006	#7-280	*23-673	*23-674	*23-675	*23-676	*23-677	*23-678	*23-689	*23-690
		*23-691	*23-692	*23-693	*23-694				
R7	-%000007	#7-280							
SAVR0	006602	*30-750	#30-757	30-807					
SAVR1	006604	*30-751	#30-758	30-808					
SAVR2	006606	*30-752	#30-759	30-809					
SAVR3	006610	*30-753	#30-760	30-810					
SAVR4	006612	*30-754	#30-761	30-811					
SAVR5	006614	*30-755	#30-762	30-812					
SAV6	004560	*23-679	23-685	#23-711					
SCOPE	= 000004	#7-280							
SDPAR0	- 172260	#7-280							
SDPAR1	= 172262	#7-280							
SDPAR2	= 172264	#7-280							
SDPAR3	= 172266	#7-280							
SDPAR4	= 172270	#7-280							
SDPAR5	= 172272	#7-280							
SDPAR6	= 172274	#7-280							
SDPAR7	- 172276	#7-280							
SDPDR0	= 172220	#7-280							
SDPDR1	= 172222	#7-280							

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
SDPDR2	= 172224	#7-280								
SDPDR3	= 172226	#7-280								
SDPDR4	= 172230	#7-280								
SDPDR5	= 172232	#7-280								
SDPDR6	= 172234	#7-280								
SDPDR7	= 172236	#7-280								
SETPAR	004672	9-354	23-687	#25-731						
SIPAR0	= 172240	#7-280	23-680	23-688	*25-731					
SIPAR1	= 172242	#7-280	*25-732							
SIPAR2	= 172244	#7-280	*25-733							
SIPAR3	= 172246	#7-280	*25-734							
SIPAR4	= 172250	#7-280								
SIPAR5	= 172252	#7-280								
SIPAR6	= 172254	#7-280								
SIPAR7	= 172256	#7-280								
SIPDR0	= 172200	#7-280								
SIPDR1	= 172202	#7-280								
SIPDR2	= 172204	#7-280								
SIPDR3	= 172206	#7-280								
SIPDR4	= 172210	#7-280								
SIPDR5	= 172212	#7-280								
SIPDR6	= 172214	#7-280								
SIPDR7	= 172216	#7-280								
SIZEHI	= 177762	#7-280								
SIZELO	= 177760	#7-280								
SR0	= 177572	#7-280	25-737	*25-737	*25-737					
SR1	= 177574	#7-280								
SR2	= 177576	#7-280								
SR3	= 172516	#7-280	*25-737	*25-737						
SSP	=%000006	#7-280								
STACK	= 001100	#7-280	7-280	7-280	7-280	8-300	10-361	10-371	10-372	10-383
		11-402	11-408	11-409	11-419	11-430	12-450	12-458	12-460	12-462
		12-472	12-483	13-501	14-512	14-517	16-541	16-550	17-561	17-568
		18-580	23-672							
STACK1	= 000500	#7-282	10-365	10-377	10-388	11-413	11-424	11-435	12-466	12-477
		12-488	13-503	14-515	15-529	16-547	17-563	21-623		
STKLMT	= 177774	#7-280								
SUPMSG	010151	9-342	#33-877							
SUPSTK	= 000700	#7-280								
SWR	= 177570	#7-297	7-298	*8-300	8-300	*8-300	*8-300	10-394	11-44	12-494
		13-506	14-519	15-536	16-551	17-570	18-582			
SWREG	000176	#7-281	8-300							
SW0	= 000001	#7-280								
SW00	= 000001	#7-280	7-280							
SW01	= 000002	#7-280	7-280							
SW02	= 000004	#7-280	7-280							
SW03	= 000010	#7-280	7-280							
SW04	= 000020	#7-280	7-280							
SW05	= 000040	#7-280	7-280							
SW06	= 000100	#7-280	7-280							
SW07	= 000200	#7-280	7-280							
SW08	= 000400	#7-280	7-280							

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
SW09	= 001000	#7-280	7-280							
SW1	= 000002	#7-280								
SW10	= 002000	#7-280								
SW11	= 004000	#7-280								
SW12	= 010000	#7-280								
SW13	= 020000	#7-280								
SW14	= 040000	#7-280	10-394	11-441	12-494	13-506	14-519	15-536	16-551	17-570
		18-582								
SW15	= 100000	#7-280								
SW2	= 000004	#7-280								
SW3	= 000010	#7-280								
SW4	= 000020	#7-280								
SW5	= 000040	#7-280								
SW6	= 000100	#7-280								
SW7	= 000200	#7-280								
SW8	= 000400	#7-280								
SW9	= 001000	#7-280								
SYSTID	= 177764	#7-280								
TBITVE	= 000014	#7-280								
TITL	006616	30-756	#30-764							
TITLE	007737	9-339	#33-873							
TKVEC	= 000060	#7-280								
TPVEC	= 000064	#7-280								
TRAPVE	= 000034	#7-280	*8-300	*8-300						
TRTVEC	000014	#7-280								
TST1	002036	#10-355	10-394							
TST10	003420	#17-552	17-570							
TST11	003552	#18-571	18-582							
TST2	002254	#11-400	11-441							
TST3	002504	#12-447	12-494							
TST4	002764	#13-495	13-506							
TST5	003070	#14-507	14-519							
TST6	003200	#15-520	15-536							
TST7	003320	#16-537	16-551							
TYPDS	= 104405	9-341	19-584	#29-745						
TYPE	= 104401	9-339	9-340	9-342	9-345	9-347	9-349	10-355	10-355	11-400
		11-400	12-447	12-447	13-495	13-495	14-507	14-507	15-520	15-520
		16-537	16-537	17-552	17-552	18-576	18-576	19-584	19-584	23-698
		23-702	26-739	27-741	28-743	#29-745	30-767	30-781	30-783	30-785
		30-797	30-799	30-800	30-805	31-817	31-819	31-821	31-823	31-825
		31-827	31-829	31-831	31-833	31-835				
TYPOC	= 104402	#29-745	30-784	32-839	32-841	32-843	32-845	32-847	32-849	32-851
		32-853	32-855	32-857						
TYPON	= 104404	#29-745								
TYPOS	= 104403	#29-745	30-782							
UDPAR0	= 177660	#7-280								
UDPAR1	= 177662	#7-280								
UDPAR2	= 177664	#7-280								
UDPAR3	= 177666	#7-280								
UDPAR4	= 177670	#7-280								
UDPAR5	= 177672	#7-280								
UDPAR6	= 177674	#7-280								

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES						
UDPAR7	= 177676	#7-280						
UDPDR0	= 177620	#7-280						
UDPDR1	= 177622	#7-280						
UDPDR2	= 177624	#7-280						
UDPDR3	= 177626	#7-280						
UDPDR4	= 177630	#7-280						
UDPDR5	= 177632	#7-280						
UDPDR6	= 177634	#7-280						
UDPDR7	= 177636	#7-280						
UIPAR0	= 177640	#7-280						
UIPAR1	= 177642	#7-280						
UIPAR2	= 177644	#7-280						
UIPAR3	= 177646	#7-280						
UIPAR4	= 177650	#7-280						
UIPAR5	= 177652	#7-280						
UIPAR6	= 177654	#7-280						
UIPAR7	= 177656	#7-280						
UIPDR0	= 177600	#7-280						
UIPDR1	= 177602	#7-280						
UIPDR2	= 177604	#7-280						
UIPDR3	= 177606	#7-280						
UIPDR4	= 177610	#7-280						
UIPDR5	= 177612	#7-280						
UIPDR6	= 177614	#7-280						
UIPDR7	= 177616	#7-280						
UPF	010037	23-699	#33-875					
USESTK	= 000600	#7-280						
USP	-%000006	#7-280						
USRPFL	010311	9-347	#33-880					
UVEC	004550	23-668	23-681	#23-709				
VIR.AD	007700	*21-617	*21-618	*21-619	21-624	32-855	33-861	#33-870
XYZ	001420	#7-296	8-332	9-343				
$APTHD	001100	7-291	#7-291					
$ASTAT	= ******	7-293	7-293					
$ATYC	001174	7-293	#7-293					
$ATY1	001150	#7-293						
$ATY3	001156	#7-293	28-743					
$ATY4	001166	#7-293						
$CHARC	006452	*28-743	*28-743	28-743	*28-743	#28-743		
$CKSWR	- ******	29-745						
$CMTAG	= ******	8-300	8-300					
$CORE	005326	25-737	#25-737					
$CPUOP	001142	#7-292						
$CRLF	006473	9-349	28-743	28-743	28-743	#28-743		
$CROUT	005356	25-737	#25-737					
$DBLK	006102	27-741	27-741	#27-741				
$DEVCT	001124	#7-292						
$DOAGN	003734	19-584	19-584	#19-584				
$DTBL	006072	27-741	#27-741					
$ENDAD	003724	7-289	9-337	#19-584				
$ENDCT	003672	8-300	#19-584					
$ENDMG	003743	19-584	#19-584					

SYMBOL CROSS REFERENCE

SYMBOL		VALUE	REFERENCES								
$ENULL		003740	10-355	11-400	12-447	13-495	14-507	15-520	16-537	17-552	18-576
			19-584	#19-584	30-785	30-799					
$ENV		001134	#7-292	7-293	7-293	28-743					
$ENVM		001135	#7-292	7-293	8-300	28-743	28-743				
$EOP		003646	#19-584								
$EOPCT		003664	*8-300	#19-584	19-584						
$ERROR		006550	7-285	12-459	#30-750						
$ETABL		001134	#7-292								
$ETEND		001150	7-291	#7-292							
$FATAL		001116	#7-292	*7-293							
$FFLG		001414	*7-293	*7-293	7-293	*7-293	#7-293				
$FILLC		006470	28-743	28-743	28-743	#28-743					
$FILLS		006467	28-743	28-743	#28-743						
$GET42		003714	#19-584								
$GTSWR	=	******	29-745								
$HD	=	000000	7-244	7-244	7-244						
$HIBTS		001100	#7-291								
$KTNEX		005320	25-737	#25-737							
$KTOUT		005256	25-737	#25-737	25-737						
$KT11		005010	*8-309	#25-737	*25-737	*25-737					
$LF		006474	28-743	28-743	#28-743						
$LFLG		001413	*7-293	#7-293							
$LSTAD		005434	*25-737	#25-737							
$LSTBK		005436	*8-311	8-312	*25-737	#25-737					
$MADR1		001146	#7-292								
$MAIL		001114	7-291	7-291	#7-292	8-300	28-743				
$MAMS1		001144	#7-292								
$MAP		005102	*25-737	*25-737	#25-737	25-737	25-737				
$MAPRG		005106	25-737	#25-737							
$MBADR		001102	#7-291								
$MFLG		001412	*7-293	7-293	*7-293	#7-293					
$MMOUT		005300	25-737	#25-737							
$MSGAD		001130	#7-292	*7-293	7-293						
$MSGLG		001132	#7-292	*7-293							
$MSGTY		001114	#7-292	7-293	*7-293	7-293	*7-293				
$MTYP1		001145	#7-292								
$NOMAP		005126	25-737	#25-737							
$NULL		006466	28-743	28-743	28-743	#28-743					
$NWTST	=	000001	#9-355	9-355	#10-355	#10-400	10-400	#11-400	#11-447	11-447	#12-447
			#12-495	12-495	#13-495	#13-507	13-507	#14-507	#14-520	14-520	#15-520
			#15-537	15-537	#16-537	#16-552	16-552	#17-552	#17-571	17-571	#18-571
$OCNT		005662	*26-739	*26-739	#26-739						
$OMODE		005664	*26-739	*26-739	26-739	*26-739	*26-739	#26-739			
$PASS		001122	#7-292	*8-300	*8-300	*19-584	*19-584	19-584	19-584	19-584	
$PASTM		001106	#7-291								
$QUES		006472	28-743	28-743	#28-743						
$RDCHR	-	******	29-745								
$RDDEC		******	29-745								
$RDLIN		******	29-745								
$RDOCT	-	******	29-745								
$RTNAD		003736	#19-584								
$R2A	-	******	29-745								

SYMBOL CROSS REFERENCE

SYMBOL		VALUE	REFERENCES								
$SAVPC	=	000204	#7-283	7-287							
$SAVRE	=	******	29-745								
$SETUP	=	000024	#7-288	7-288	#7-288	7-288	#7-288	8-300	8-300	8-300	8-300
			8-300	8-300	8-300	8-300	8-300	8-300	8-300	8-300	19-584
			19-584								
$SIZE		004724	8-310	#25-737							
$SIZEX		005362	25-737	25-737	#25-737						
$STOP		005104	*25-737	*25-737	#25-737	25-737					
$STUP	=	177777	#7-288	#7-288	7-288	#7-288	#7-288	7-288			
$SVPC	=	000204	#7-289	7-289							
$SWR	=	040000	#7-241	7-244	7-247	7-247	7-247	7-247	7-247	7-247	7-247
			7-247	8-300	10-355	11-400	12-447	13-495	14-507	15-520	16-537
			17-552	18-571	19-584	19-584	19-584	19-584	19-584		
$SWREG		001136	#7-292	8-300							
$TESTN		001120	#7-292	*9-348	*11-401	*12-448	*13-496	*14-508	*15-521	*16-538	*17-553
			*18-572	30-782							
$TKB		006460	28-743	28-743	28-743	28-743	#28-743				
$TKS		006456	28-743	28-743	28-743	28-743	#28-743				
$TN	=	000012	#7-242	7-244	9-355	10-355	10-355	#10-355	10-394	10-400	11-400
			11-400	#11-400	11-441	11-447	12-447	12-447	#12-447	12-494	12-495
			13-495	13-495	#13-495	*13-506	13-507	14-507	14-507	#14-507	*14-519
			14-520	15-520	15-520	#15-520	15-536	15-537	16-537	16-537	#16-537
			*16-551	16-552	17-552	17-552	#17-552	*17-570	17-571	18-571	#18-571
			*18-576	18-582							
$TPB		006464	28-743	28-743	28-743	#28-743					
$TPFLG		006471	28-743	28-743	28-743	#28-743					
$TPS		006462	28-743	28-743	28-743	#28-743					
$TRAP		006476	8-300	#29-745							
$TRAP2		006520	#29-745	29-745							
$TRP	-	000006	#29-745	29-745	29-745	29-745	29-745	#29-745	29-745	29-745	29-745
			29-745	#29-745	29-745	29-745	29-745	29-745	#29-745	29-745	29-745
			29-745	29-745	#29-745	29-745	29-745	29-745	29-745	#29-745	
$TRPAD		006532	29-745	#29-745							
$TSTM		001104	#7-291								
$TYPBN	=	******	29-745								
$TYPDS		005666	#27-741	29-745	29-745						
$TYPE		006112	7-293	#28-743	29-745	29-745					
$TYPEC		006324	28-743	28-743	28-743	#28-743					
$TYPEX		006454	28-743	28-743	28-743	#28-743					
$TYPOC		005464	#26-739	29-745	29-745						
$TYPON		005500	26-739	#26-739	29-745						
$TYPOS		005440	#26-739	29-745							
$UNIT		001126	#7-292								
$UNITM		001110	#7-291								
$USWR		001140	#7-292								
$XOFF	=	000023	28-743	28-743							
$XON	=	000021	28-743	28-743	28-743						
$$GET4	=	000000	#19-584	19-584							
$OFILL		005663	*26-739	*26-739	26-739	#26-739					
.$ASTA		******	7-293	7-293							
.$X		001100	#7-291	7-291							

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES									
COMMEN	#7-280									
ENDCOM	#7-280									
ERR	#7-255	#10-365	#10-377	#10-388	#11-413	#11-424	#11-435	#12-466	#12-477	#12-488
	#13-503	#14-515	#15-529	#16-547	#17-563	#21-623				
ESCAPE	#7-280									
GETPRI	#7-280	25-737								
GETSWR	#7-280									
LOOP	#7-273	10-394	11-441	12-494	13-506	14-519	15-536	16-551	17-570	18-582
LOOPX	#7-276	10-394	11-441	12-494	13-506	14-519	15-536	16-551	17-570	18-582
MULT	#7-280									
NEWTST	#7-280	#9-355	#10-400	#11-447	#12-495	#13-507	#14-520	#15-537	#16-552	#17-571
PFAIL	#7-249	#10-358	#11-405	#12-454	#13-500	#14-511	#15-527	#16-542	#17-559	#18-578
POP	#7-280	#7-293	#7-293	#27-741						
PUSH	#7-280	#7-293	#7-293	#7-293	#27-741					
REPORT	#7-280									
SETK	#7-270	10-356	12-449	13-497	14-513	15-522	16-539	17-554	18-573	
SETPRI	#7-280									
SETTRA	#29-745	#29-745	#29-745	#29-745	#29-745	#29-745				
SETUP	#7-280	#8-300								
SKIP	#7-280									
SLASH	#7-280									
SPACE	#7-252	#7-280								
STARS	#7-280	#7-289	#7-291	#7-291	#7-291	#7-292	#7-293	#10-355	#10-355	#11-400
	#11-400	#12-447	#12-447	#13-495	#13-495	#14-507	#14-507	#15-520	#15-520	#16-537
	#16-537	#17-552	#17-552	#18-571	#18-571	#19-584	#23-665	#25-737	#26-739	#27-741
	#28-743	#29-745								
SWRSU	#7-280	#8-300	8-300							
TEST	#7-263	#10-355	#11-400	#12-447	#13-495	#14-507	#15-520	#16-537	#17-552	#18-576
TRMTRP	#29-745									
TSTNUM	#7-266	#10-355	#11-400	#12-447	#13-495	#14-507	#15-520	#16-537	#17-552	#18-576
TYPBIN	#7-280									
TYPDEC	#7-280	#9-341	#19-584							
TYPNAM	#7-280									
TYPNUM	#7-280									
TYPOCS	#7-280	30-782								
TYPOCT	#7-280	30-784	32-839	32-841	32-843	32-845	32-847	32-849	32-851	32-853
	32-855	32-857								
TYPTXT	#7-280	#10-355	#11-400	#12-447	#13-495	#14-507	#15-520	#16-537	#17-552	#18-576
	#30-767	#30-781	#30-783	#30-797	#30-800	#30-805	#31-817	#31-819	#31-821	#31-823
	#31-825	#31-827	#31-829	#31-831	#31-833	#31-835				
$$ESCA	#7-280									
$$NEWT	#7-280	9-355	10-400	11-447	12-495	13-507	14-520	15-537	16-552	17-571
$$SET	#29-745	#29-745	#29-745	#29-745	#29-745	#29-745				
$$SETM	#8-300	8-300								
$$SKIP	#7-280									
.EQUIV	#7-235	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280
	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280
	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280
	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280
	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280	7-280
	7-280	7-280	7-280							
.HEADE	#7-238	#7-244								

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES		
.SETUP	#7-238	#7-239	#7-288
.SWRHI	#7-238	#7-247	
.SWRLO	#7-247		
.$ACT1	#7-239	#7-289	
.$APTB	#7-239	7-292	
.$APTH	#7-239	7-291	
.$APTV	#7-239	7-293	
.$CATC	#7-238	#7-281	
.$EOP	#7-238	19-584	
.$POWE	#7-238		
.$SIZE	#7-239	25-737	
.$TRAP	#7-239	#29-745	
.$TYPD	#7-239	#27-741	
.$TYPE	#7-238	28-743	
.$TYPO	#7-238	26-739	
.1170	#7-238	#7-280	