

.REM %

IDENTIFICATION  
-----

PRODUCT CODE:	AC-F615C-MC
PRODUCT NAME:	CKKACC0 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 SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1979, 1981 BY DIGITAL EQUIPMENT CORPORATION

1  
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

CONTENTS

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

- 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

83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127

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.

128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179

5. OPERATING PROCEDURE

5.1 MANUAL

START PROGRAM AT ADDRESS 200.  
A MESSAGE WILL BE TYPED IDENTIFYING 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 SUPPRESSES 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

181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228

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

244

```

.*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.
.*

```

245  
246  
247

```

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

```

248  
280

```

.SBTTL BASIC DEFINITIONS
.*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
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                    ;;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) VECTOR
000114 CACHVEC=114 ;;CACHE ERROR INTERRUPT VECTOR
000240 PIQVEC=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 MAINTENANCE REGISTER
177752 HITMIS = 177752 ;;HIT MISS REGISTER '1' IMPLIFIS 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 REGISTERS
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 MAPLOC = 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
```

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
28' .SBTTL TRAP CATCHER
      .=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
      .=174
000174 DISPREG: .WORD 0           ;;SOFTWARE DISPLAY REGISTER
000176 SWREG:  .WORD 0           ;;SOFTWARE SWITCH REGISTER
      .SBTTL STARTING ADDRESS(ES)
000200 JMP @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
282 STACK1=500
283 $SAVPC=.
284 .=30
285 $ERROR
286 340
287 .=$SAVPC
289 .SBTTL ACT11 HOOKS
      ;*****
      ;HOOKS REQUIRED BY ACT11
      $SVPC=.           ;SAVE PC
000046 .=46
      $ENDAD           ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052 .=52
000052 .WORD 0           ;;2)SET LOC.52 TO ZERO
      .=$SVPC           ;; RESTORE PC
290 .-1100
291 .SBTTL APT PARAMETER BLOCK
      ;*****
      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
      ;*****
000024 .$.X=.           ;;SAVE CURRENT LOCATION
000024 .=24           ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000044 200           ;;FOR APT START UP
000044 .=44           ;;POINT TO APT INDIRECT ADDRESS PNTR.
      $APTHDR           ;;POINT TO APT HEADER BLOCK
000044 .=$X           ;;RESET LOCATION COUNTER
      ;*****
      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
      ;INTERFACE SPEC.
001100 $APTHD:
001100 $HIBTS: .WORD 0           ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001102 $MADR: .WORD $MAIL           ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001104 $STMT: .WORD 5           ;;RUN TIM OF LONGEST TEST
001106 $PASTM: .WORD 5           ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001110 $UNITM: .WORD 0           ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
    
```

292 001112 000016

.WORD \$ETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE (WORDS)  
.SBTTL APT MAILBOX-ETABLE

\*\*\*\*\*

001114  
001114 000000  
001116 000000  
001120 000000  
001122 000000  
001124 000000  
001126 000000  
001130 000000  
001132 000000  
001134  
001134 000  
001135 000  
001136 000000  
001140 000000  
001142 000000

```

.EVEN
$MAIL:                ;;APT MAILBOX
$MSGTY: .WORD  AMSGTY  ;;MESSAGE TYPE CODE
$FATAL: .WORD  AFATAL  ;;FATAL ERROR NUMBER
$TESTN: .WORD  ATESTN  ;;TEST NUMBER
$PASS:  .WORD  APASS   ;;PASS COUNT
$DEVCT: .WORD  ADEVCT  ;;DEVICE COUNT
$UNIT:  .WORD  AUNIT   ;;I/O UNIT NUMBER
$MSGAD: .WORD  AMSGAD  ;;MESSAGE ADDRESS
$MSGLG: .WORD  AMSGLG  ;;MESSAGE LENGTH
$ETABLE:                ;;APT ENVIRONMENT TABLE
$ENV:   .BYTE  AENV    ;;ENVIRONMENT BYTE
$ENVM:  .BYTE  AENVM   ;;ENVIRONMENT MODE BITS
$SWREG: .WORD  ASWREG  ;;APT SWITCH REGISTER
$USWR:  .WORD  AUSWR   ;;USER SWITCHES
$CPUOP: .WORD  ACPUOP  ;;CPU TYPE, OPTIONS
                BIT 15=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
$MAMS1: .BYTE  AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
$MTYP1: .BYTE  AMTYP1  ;;MEM. TYPE,BLK#1
                MEM.TYPE BYTE  -- (HIGH BYTE)
                900 NSEC CORE=001
                300 NSEC BIPOLAR=002
                500 NSEC MOS=003
$MADR1: .WORD  AMADR1  ;;HIGH ADDRESS,BLK#1
                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
    
```

001144 000  
001145 000

001146 000000

001150

\$ETEND:  
.MEXIT

293

.SBTTL APT COMMUNICATIONS ROUTINE

\*\*\*\*\*

001150 112767 000001 000236  
001156 112767 000001 000226  
001164 000403  
001166 112767 000001 000220  
001174  
001174 010046  
001176 010146  
001200 105767 000206  
001204 001450  
001206 122767 000001 177720  
001214 001031  
001216 132767 000100 177711  
001224 001425  
001226 017600 000004  
001232 062766 000002 000004  
001240 005767 177650  
001244 001375  
001246 010067 177656  
001252 105720  
001254 001376

```

$ATY1:  MOV  #1,$FFLG  ;;TO REPORT FATAL ERROR
$ATY3:  MOV  #1,$MFLG  ;;TO TYPE A MESSAGE
        BR   $ATYC
$ATY4:  MCVB #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
$ATYC:
        MOV  R0,-(SP)  ;;PUSH R0 ON STACK
        MOV  R1,-(SP)  ;;PUSH R1 ON STACK
        TSTB $MFLG    ;;SHOULD TYPE A MESSAGE?
        BEQ  5$        ;;IF NOT: BR
        CMPB #APTENV,$ENV  ;;OPERATING UNDER APT?
        BNE  3$        ;;IF NOT: BR
        BITB #APTPOOL,$ENVM  ;;SHOULD SPOOL MESSAGES?
        BEQ  3$        ;;IF NOT: BR
        MOV  @4(SP),R0  ;;GET MESSAGE ADDR.
        ADD  #2,4(SP)  ;;BUMP RETURN ADDR.
1$:     TST  $MSGTYPE  ;;SEE IF DONE W/ LAST XMISSION?
        BNE  1$        ;;IF NOT: WAIT
        MOV  R0,$MSGAD  ;;PUT ADDR IN MAILBOX
2$:     TSTB (R0)+    ;;FIND END OF MESSAGE
        BNE  2$
    
```

001256	166700	177646		SUB	\$MSGAD,RO	::SUB START OF MESSAGE
001262	006200			ASR	R0	::GET MESSAGE LNTH 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
001413	000			\$LFLG:	.BYTE	0
001414	000			\$FFLG:	.BYTE	0
					.EVEN	
	000200			APTSIZE=	200	
	000001			APTENV=	001	
	000100			APTSPool=	100	
	000040			APTCsUP=	040	
294				PFAIL:	.WORD	0
295	001416	000000		XYZ:	.WORD	0
296	001420	000000		SWR=	177570	
297		177570		DISPLAY=	SWR	
298		177570		LRTI:	RTI	
299	001422	000002				



CKKACCO 11/44 POWER FAIL  
INITIALIZE THE COMMON TAGS

MACRO M1113 06-APR-81 14:07 PAGE 8-1<sup>D 2</sup>

SEQUENCE 16

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	MOV	#6,4	:RESTORE TRAP VEC
334	001742	005067	176040		CLR	6	



```

335                               ;IDENTIFY THE PROGRAM AND TYPE INSTRUCTIONS
336
337 001746 023767 000042 001750      CMP      @#42,$ENDAD
338 001754 001421                BEQ      PRETST
339 001756 104401 007737          TYPE     ,TITLE           ;TYPE THE TITLE
340 001762 104401 007774          TYPE     ,BOOT
341 001766 010046                MOV      RO,-(SP)           ;;SAVE RO FOR TYPEOUT
    001770 104405                TYPDS
    001772 104401 010151          TYPE     ,SUPMSG           ;;GO TYPE--DECIMAL ASCII WITH SIGN
342 001772 104401 010151          TYPE     ,SUPMSG           ;TYPE THE STARTUP MESSAGE
343 001776 026727 177414 001420    CMP      PFAIL,#XYZ       ;IS AUTOMATIC POWER FAIL
344 002004 001403                BEQ      5$              ;BRANCH IF NO
345 002006 104401 010371          TYPE     ,HDWPFL         ;TYPE HARDWARE POWER FAILURE MESSAGE
346 002012 000402                BR       PRETST         ;BRANCH OVER MANUAL MESSAGE
347 002014 104401 010311          5$:     TYPE     ,USRPF1         ;TYPE USER POWER FAIL MESSAGE
348 002020 012767 000001 177072    PRETST: MOV    #1,$TESTN
349 002026 104401 006473          TYPE     ,$CRLF         ;TYPE A <CRLF>
350                               ;THIS ROUTINE IS FOR CHECKING 2MS TIMING USING SOB OUT OF SIPARS
351                               ;BRANCH . OUT OF SIPARO=1.2US
352                               ;SOB HAS ONE MORE MICRO STEP THROUGH EQUAL 1.4US
353                               ;1024 MUL BY 1.4US EQUAL 1.43MS MIN TIME CHECKED FOR ILLUP,ILLDWN
354 002032 004767 002634          JSR      PC,SETPAR       ;GO PUT TIME FACTOR SUBROUTINE IN SIPARO-SIPAR3

```

355

```
.SBTTL TEST # 1 - SIMPLE DOWN/UP TEST (KERNEL)
:*****
:*TEST 1 SIMPLE DOWN/UP TEST (KERNEL)
:*****
TST1:
```

002036  
002036 104401 002044  
002042 000402  
  
002050  
002050 104401 003740  
356 002054 005037 177776  
357 002060 012703 002074  
358 002064 012777 000001 17/324  
359 002072 000001  
360 002074 010600  
361 002076 022700 001074  
362 002102 001414  
363 002104 012767 001074 005556  
364 002112 010067 005554  
365 002116 012706 000500  
  
002122 104000

```
TYPE ,65$ ;:TYPE ASCIZ STRING  
BR 64$ ;:GET OVER THE ASCIZ  
:65$: .ASCIZ *1 *  
64$:  
TYPE , $ENULL  
CLR @#PS ;:SET KERNEL MODE  
MOV #2$,R3 ;:SET POWER UP RETURN  
MOV #1,@PFAIL ;:SET UP TO START POWER FAIL  
WAIT ;:WAIT FOR POWER FAIL  
2$: MOV SP,R0 ;:GET SP  
CMP #STACK-4,R0 ;:CHECK SP  
BEQ 3$ ;:SKIP IF OK  
MOV #1074,EXP.SP1  
MOV R0,REC.SP1  
MOV #STACK1,SP ;:SETUP RESERVED STACK POINTER  
;:FOR ERROR PRINTOUT  
  
ERROR  
  
;ERROR  
;-----
```

366  
367 002124 007670  
368 002126 007672  
369 002130 000000  
370 002132 000000  
371 002134 012706 001100  
372 002140 013700 001074  
373 002144 022700 002074  
374 002150 001414  
375 002152 012767 002074 005504  
376 002160 010067 005502  
377 002164 012706 000500  
  
002170 104000

```
EXP.SP1 ;:SP NOT STACK-4  
REC.SP1 ;:PRINT EXPECTED STACK POINTER  
0 ;:PRINT RECEIVED STACK POINTER  
HALT  
3$: MOV #STACK,SP ;:RESET SP  
MOV @#STACK-4,R0 ;:GET RETURN ADDRESS  
CMP #2$,R0 ;:CHECK ADDRESS  
BEQ 4$ ;:SKIP IF OK  
MOV #2$,EXP.ADD  
MOV R0,REC.ADD  
MOV #STACK1,SP ;:SETUP RESERVED STACK POINTER  
;:FOR ERROR PRINTOUT  
  
ERROR  
  
;ERROR  
;-----
```

378  
379 002172 007664  
380 002174 007666  
381 002176 000000  
382 002200 000000  
383 002202 013700 001076  
384 002206 022700 000000  
385 002212 001414  
386 002214 012767 000000 005452  
387 002222 010067 005450  
388 002226 012706 000500  
  
002232 104000

```
EXP.ADD ;:ADDRESS ON STACK IS WRONG  
REC.ADD ;:PRINT EXPECTED ADDRESS  
0 ;:PRINT RECEIVED ADDRESS  
HALT  
4$: MOV @#STACK-2,R0 ;:GET OLD PS  
CMP #0,R0 ;:CHECK OLD PS  
BEQ 5$ ;:SKIP IF OK  
MOV #0,EXP.PSW  
MOV R0,REC.PSW  
MOV #STACK1,SP ;:SETUP RESERVED STACK POINTER  
;:FOR ERROR PRINTOUT  
  
ERROR
```

;ERROR

```
389                                     ;-----  
390 002234 007674                       ;OLD PS IS WRONG  
391 002236 007676                       ;PRINT EXPECTED PS  
392 002240 000000                       ;PRINT RECEIVED PS  
393 002242 000000  
394 002244 000000  
      002244 032737 040000 177570 5$:   ;  
      002252 001271                       ;  
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
*****
```

002254  
 002254 104401 002262  
 002260 000402

```
TST2:
      TYPE      .65$      ;:TYPE ASCIZ STRING
      BR        64$      ;:GET OVER THE ASCIZ
::65$: .ASCIZ *2 *
64$:
```

002266  
 002266 104401 003740  
 401 002272 005267 176622  
 402 002276 012706 001100  
 403 002302 012767 000357 175466  
 404 002310 012703 002324  
 405 002314 012777 000001 177074  
 406 002322 000777  
 407 002324 010600  
 408 002326 012706 001100  
 409 002332 022700 001074  
 410 002336 001414  
 411 002340 012767 001074 005322  
 412 002346 010067 005320  
 413 002352 012706 000500

```
      TYPE,$ENULL
      INC      $TESTN
      MOV      #STACK,SP      ;SET UP STACK
      MOV      #357,PS      ;SET UP CONDITION CODES
      MOV      #2$,R3      ;SET POWER UP RETURN
      MOV      #1,@PFFAIL    ;SET UP TO START POWER FAIL
1$: BR
2$: MOV      SP,R0      ;WAIT FOR POWER FAIL
      MOV      #STACK,SP    ;GET SP
      CMP      #STACK-4,R0  ;SET STACK
      BEQ      3$          ;CHECK STACK POINTER
      MOV      #1074,EXP.SP1 ;SKIP IF OK
      MOV      R0,REC.SP1
      MOV      #STACK1,SP   ;SETUP RESERVED STACK POINTER
                               ;FOR ERROR PRINTOUT
```

002356 104000

ERROR

;ERROR

414  
 415 002360 007670  
 416 002362 007672  
 417 002364 000000  
 418 002366 000000  
 419 002370 013700 001076  
 420 002374 022700 000341  
 421 002400 001414  
 422 002402 012767 000341 005264  
 423 002410 010067 005262  
 424 002414 012706 000500

```
      EXP.SP1
      REC.SP1
      O
      HALT
3$: MOV      @#STACK-2,R0    ;GET OLD PS
      CMP      #341,R0      ;CHECK OLD PS
      BEQ      4$          ;SKIP IF OK
      MOV      #341,EXP.PSW
      MOV      R0,REC.PSW
      MOV      #STACK1,SP   ;SETUP RESERVED STACK POINTER
                               ;FOR ERROR PRINTOUT
```

002420 104000

ERROR

;ERROR

425  
 426 002422 007674  
 427 002424 007676  
 428 002426 000000  
 429 002430 000000  
 430 002432 0 ,700 001074  
 431 002436 022700 002322  
 432 002442 001414  
 433 002444 012767 002322 005212  
 434 002452 010067 005210  
 435 002456 012706 000500

```
      EXP.PSW
      REC.PSW
      O
      HALT
4$: MOV      @#STACK-4,R0    ;GET RETURN ADDRESS
      CMP      #1$,R0
      BEQ      5$          ;SKIP IF OK
      MOV      #1$,EXP.ADD
      MOV      R0,REC.ADD
      MOV      #STACK1,SP   ;SETUP RESERVED STACK POINTER
                               ;FOR ERROR PRINTOUT
```

002462 104000

ERROR

;SETUP RESERVED STACK POINTER  
 ;FOR ERROR PRINTOUT

436  
437 002464 007664  
438 002466 007666  
439 002470 000000  
440 002472 000000  
441 002474  
002474 032737 040000 177570  
002502 001264

5\$:

EXP.ADD  
REC.ADD  
0  
HALT

BIT #SW14,@#SWR  
BNE TST2

:LOOP ON TEST?  
:LOOP TO TST2

442  
443  
444  
445  
446

:TEST ROUTINE TO CHECK RESTART CAPABILITY  
:USING THE EMULATOR TRAP FOR A WAIT  
:OPERATOR MUST SET HALT SWITCH TO ENABLE POSITION  
:

:ERROR  
:-----  
:ADDRESS ON STACK IS WRONG  
:PRINT EXPECTED ADDRESS  
:PRINT RECEIVE ADDRESS

447

.SBTTL TEST # 3 - POWER FAIL WITH EMT TRAP  
:\*\*\*\*\*  
:\*TEST 3 POWER FAIL WITH EMT TRAP  
:\*\*\*\*\*  
TST3:

002504  
002504 104401 002512  
002510 000402  
  
002516  
002516 104401 003740  
448 002522 005267 176372  
449 002526 005037 177776  
450 002532 012706 001100  
451 002536 012703 002570  
452 002542 012767 001422 175260  
453 002550 012767 000005 175254  
454 002556 012777 000001 176632  
455 002564 104000  
456 002566 000776  
457 002570 010600  
458 002572 012706 001100  
459 002576 012767 006550 175224  
460 002604 022700 001074  
461 002610 00146  
462 002612 022700 001070  
463 002616 001414  
464 002620 012767 001070 005042  
465 002626 010067 005040  
466 002632 012706 000500

TYPE ,65\$ ;:TYPE ASCIZ STRING  
BR 64\$ ;:GET OVER THE ASCIZ  
:65\$: .ASCIZ \*3 \*  
64\$:  
TYPE ,SENULL  
INC \$TESTN  
CLR @#PS ;SET KERNEL MODE  
MOV #STACK,SP ;SET UP STACK  
MOV #2\$,R3 ;SET POWER UP RETURN  
MOV #LRTI,EMTVEC ;SET UP RETURN FROM EMT TRAP VECTOR  
MOV #5,EMTVEC+2 ;SET PSW ON POWER FAIL  
MOV #1,@PFAIL ;SET UP TO START POWER FAIL  
3\$: EMT ;WAIT FOR POWER FAIL  
BR 3\$  
2\$: MOV SP,R0 ;GET SP  
MOV #STACK,SP  
MOV #ERROR,30 ;RESET EMT VECTOR  
CMP #STACK-4,R0 ;WAS STACK PUSHED ONLY TWICE  
BEQ 6\$  
CMP #STACK-10,R0 ;WAS STACK PUSHED 4 TIMES  
BEQ 4\$  
MOV #1070,EXP.SP1  
MOV R0,REC.SP1  
MOV #STACK1,SP ;SETUP RESERVED STACK POINTER  
;FOR ERROR PRINTOUT

002636 104000  
  
467  
468 002640 007670  
469 002642 007672  
470 002644 000000  
471 002646 000000  
472 002650 013700 001070  
473 002654 022700 001422  
474 002660 001414  
475 002662 012767 001422 004774  
476 002670 010067 004772  
477 002674 012706 000500

ERROR  
  
;ERROR  
;-----  
;WAS STACK PUSHED 4 TIMES  
;PRINT EXPECTED STACK POINTER  
;PRINT RECEIVED STACK POINTER  
4\$: MOV @#STACK-10,R0  
CMP #LRTI,R0 ;DOES STACK CONTAIN CORRECT INFO  
BEQ 5\$  
MOV #LRTI,EXP.ADD  
MOV R0,REC.ADD  
MOV #STACK1,SP ;SETUP RESERVED STACK POINTER  
;FOR ERROR PRINTOUT

002700 104000  
  
478  
479 002702 007664  
480 002704 007666  
481 002706 000000  
482 002710 000000  
483 002712 013700 001072  
484 002716 022700 000005

ERROR  
  
;ERROR  
;-----  
;IS STACK PUSHED MORE THAN 2 OR LESS THAN 4  
;PRINT EXPECTED ADDRESS  
;PRINT RECEIVED ADDRESS  
5\$: MOV @#STACK-6,R0  
CMP #5,R0 ;DOES STACK CONTAIN CORRECT INFO

485 002722 001414  
486 002724 012767 000005 004742  
487 002732 000000 007676  
488 002736 012706 000500  
  
002742 104000

BEG 6\$  
MOV #5,EXP.PSW  
RO,REC.PSW  
MOV #STACK1,SP  
  
ERROR

; SETUP RESERVED STACK POINTER  
; FOR ERROR PRINTOUT

489  
490 002744 007674  
491 002746 007676  
492 002750 000000  
493 002752 000000  
494 002754  
002754 032737 040000 177570 6\$:  
002762 001250

EXP.PSW  
REC.PSW  
0  
HALT  
  
BIT #SW14,@#SWR  
BNE TST3

; ERROR  
;-----  
; STACK CONTIAN WRONG PSW  
; PRINT EXPECTED PSW  
; PRINT RECEIVED PSW

; LOOP ON TEST?  
; LOOP TO TST3

495

```
.SBTTL TEST # 4 - POWER FAIL WITH ODD ADDRESS
:*****
:*TEST 4 POWER FAIL WITH ODD ADDRESS
:*****
```

```
002764
002764 104401 002772
002770 000402

002776
002776 104401 003740
496 003002 005267 176112
497 003006 005037 177776
498 003012 012737 003032 000004
499 003020 012703 003052
500 003024 012777 000001 176364
501 003032 012706 001100
502 003036 005737 000003
503 003042 012706 000500

003046 104000

504 003050 000000
505 003052 012737 000006 000004
506 003060 032737 040000 177570
003066 001336
```

```
TST4:
TYPE ,65$ ;:TYPE ASCIZ STRING
BR ,4$ ;:GET OVER THE ASCIZ
;:65$: .ASCIZ *4 *
64$:
TYPE , $ENUL
INC $TESTN ;:SET KERNEL MODE
CLR @#PS ;:SET TRAP VECTOR
MOV #3$, @#ERRVEC ;:SET RETURN ADDRESS FOR POWER FAIL
MOV #1$, R3 ;:SET UP TO START POWER FAIL
MOV #1, @PFAIL ;:RESET STACK
3$: MOV #STACK, SP ;:CAUSE ODD ADDRESS TRAP
TST @#3 ;:SETUP RESERVED STACK POINTER
MOV #STACK1, SP ;:FOR ERROR PRINTOUT

ERROR

;:ERROR
:-----
:ODD ADDRESS TRAP FAILED TO OCCUR
1$: MOV #6, @#ERRVEC ;:RESET 4
BIT #SW14, @#SWR ;:LOOP ON TEST?
BNE TST4 ;:LOOP TO TST4
```



507

```
.SBTTL TEST # 5 - POWER FAIL WITH TIME OUT (KERNEL)
*****
;*TEST 5 POWER FAIL WITH TIME OUT (KERNEL)
*****
```

```
003070
003070 104401 003076
003074 000402
```

```
TST5:
TYPE ,65$ ;:TYPE ASCIZ STRING
BR 64$ ;:GET OVER THE ASCIZ
::65$: .ASCIZ *5 *
```

```
003102
508 003106 104401 003740
003106 005267 176006
509 003112 012737 003132 000004
510 003120 012703 003156
511 003124 012777 000001 176264
512 003132 012706 001100
513 003136 005037 177776
514 003142 010037 160000
515 003146 012706 000500
```

```
64$:
TYPE,$ENULL
INC $TESTN
MOV #3$,@#ERRVEC ;:SET TRAP VECTOR
MOV #1$,R3 ;:SET UP RETURN ADDRESS FOR POWER FAIL
MOV #1,@PFAIL ;:SET UP TO START POWER FAIL
3$: MOV #STACK,SP ;:SET STACK
CLR @#PS ;:SET KERNEL MODE
MOV RO,@#160000 ;:CAUSE A TIMEOUT
MOV #STACK1,SP ;:SETUP RESERVED STACK POINTER
;:FOR ERROR PRINTOUT
```

```
003152 104000
```

```
ERROR
```

```
;:ERROR
```

```
516 003154 000000
517 003156 012706 001100
518 003162 012737 000006 000004
519 003170 032737 040000 177570
003176 001334
```

```
1$: HALT ;:TIME OUT FAILED TO OCCUR
MOV #STACK,SP ;:SET STACK
MOV #6,@#ERRVEC ;:RESET 4
BIT #SW14,@#SWR ;:LOOP ON TEST?
BNE TST5 ;:LOOP TO TST5
```

520

```
.SBTTL TEST # 6 - POWER FAIL IN THE STACK OVERFLOW (KERNEL)
:*****
:*TEST 6 POWER FAIL IN THE STACK OVERFLOW (KERNEL)
:*****
TST6:
```

```
003200 003200 104401 003206      TYPE ,65$           ;;TYPE ASCIZ STRING
003204 000402                   BR 64$             ;;GET OVER THE ASCIZ
:65$: .ASCIZ *6 *
64$:
003212 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$,R3     ;SET RETURN
534 003304 000002      RTI           ;GO TO THE POWER FAIL ROUTINE
535 003306 000000      HALT
536 003310 003310      4$:
003310 032737 040000 177570      BIT #SW14,@#SWR ;LOOP ON TEST?
003316 001330      BNE TST6       ;LOOP TO TST6
```

.J37

```
.SBTTL TEST # 7 - POWER FAIL WITH RESETS
:-----:
:TEST 7      POWER FAIL WITH RESETS
:-----:
TST7:
```

```

003320 104401 003326
003324 000402
003332
003332 104401 003740
538 003336 005267 175556
539 003342 005037 177776
540 003346 012703 003404
541 003352 012706 001100
542 003356 012777 000001 176032
543 003364 000005
544 003366 000005
545 003370 000005
546 003372 000774
547 003374 012706 000500
003400 104000

                                TYPE      .65$      ;;TYPE ASCIZ STRING
                                BR        64$      ;;GET OVER THE ASCIZ
::65$: .ASCIZ *7 *
64$:
                                TYPE, $ENULL
                                INC      $TESTN
                                CLR      @#PS      ;SET KERNEL MODE
                                MOV      #1$,R3    ;SET RETURN ADDRESS
                                MOV      #STACK,SP ;RESET STACK
                                MOV      #1,@PFAIL  ;SET UP TO START POWER FAIL
3$:  RESET                       ;RESETS
      RESET                       ;TO WAIT
      RESET                       ;IN
      BR        3$                 ;LOOP
      MOV      #STACK1,SP        ;SETUP RESERVED STACK POINTER
                                ;FOR ERROR PRINTOUT
                                ;ERROR
                                ;-----
                                ;RESFT LOOP FAILED
548
549 003402 000000
550 003404 012706 001100
551 003410 032737 040000 177570
003416 001340
1$:  HALT
      MOV      #STACK,SP        ;RESET STACK
      BIT      #SW14,@#SWR     ;LOOP ON TEST?
      BNE     TST7             ;LOOP TO TST7
```

552

```
.SBTTL TEST # 10 - MEMORY MANAGEMENT ABORT TEST
:*****
:*TEST 10 MEMORY MANAGEMENT ABORT TEST
:*****
TST10:
```

003420  
 003420 104401 003426  
 003424 000402

```
TYPE ,65$ ;:TYPE ASCII STRING
BR ,64$ ;:GET OVER THE ASCII
:65$: .ASCII *10 *
```

003432  
 003432 104401 003740  
 553 003436 005267 175456  
 554 003442 005037 177776  
 555 003446 012737 003522 000004  
 556 003454 004767 001110  
 557 003460 012737 003504 000250  
 558 003466 012703 003524  
 559 003472 012777 000001 175716  
 560 003500 005237 177572  
 561 003504 012706 001100  
 562 003510 005237 140C00  
 563 003514 012706 000500

```
64$: TYPE,$ENULL
INC $TESTN
CLR @#PS ;:SET KERNEL MODE
MOV #4$,@#ERRVEC ;:SET FOR TIMEOUT
JSR PC,MAP ;MAP THE WORLD
MOV #3$,@#MMVEC ;:SET MEMORY MANAGEMENT VECTOR
MOV #1$,R3 ;:LOAD PF RETURN
MOV #1,@PFAIL ;:SET UP TO START POWER FAIL
INC @#MMRO ;:TURN MEMORY MANAGEMENT ON
3$: MOV #STACK,SP ;:ZAP STACK
INC @#1400C0 ;:ACCESS VIOLATION
MOV #STACK1,SP ;:SETUP RESERVED STACK POINTER
;:FOR ERROR PRINTOUT
```

003520 104000

ERROR

;:ERROR

;:NO VIOLATION OR TRAP TO 4

564  
 565 003522 000000  
 566  
 567 003524 005037 177572  
 568 003530 012706 001100  
 569 003534 012737 000006 000004  
 570 003542 032737 040000 177570  
 003550 001323

```
4$: HALT
1$: CLR @#MMRO ;:TURN OFF MEMORY MANAGEMENT
2$: MOV #STACK,SP ;:MAKE A NEW STACK
MOV #6,@#ERRVEC ;:RESET 4
BIT #SW14,@#SWR ;:LOOP ON TEST?
BNE TST10 ;:LOOP TO TST10
```

571

.SBTTL TEST # 11 - MEMORY VOLATILITY TEST

\*\*\*\*\*  
 :\*TEST 11 MEMORY VOLATILITY TEST  
 \*\*\*\*\*

003552 000240  
 572 003554 005267 175340  
 573 003560 005037 177776  
 574 003564 004767 000170  
 575 003570 012703 003626  
 576 003574 104401 003602  
 003600 000402

```
TST11: NOP
        INC     $TESTN
        CLR     @#PS           ;SET KERNEL MODE
4$:     JSR     PC,LOAD        ;LOAD ALL MEMORY WITH 52525
        MOV     #1$,R3       ;POWER FAIL RETURN ADDRESS
        TYPE   ,65$         ;;TYPE ASCIZ STRING
        BR     64$          ;;GET OVER THE ASCIZ
        ;;65$: .ASCIZ *11 *
        64$:
        TYPE, $ENULL
2$:     JSR     PC,CHECK      ;CHECK FOR THE 52525
        MOV     #1,@PFAIL    ;SET UP TO START POWER FAIL
        BR     2$           ;LOOP FOR EVER OR POWER FAIL
1$:     MOV     #STACK,SP    ;ZAP THE STACK
        JSR     PC,CHECK      ;CHECK ALL MEMORY
        BIT     #SW14,@#SWR  ;LOOP ON TEST?
        BNE    TST11        ;LOOP TO TST11
```

003606  
 003606 104401 003740  
 577 003612 004767 000246  
 578 003616 012777 000001 175572  
 579 003624 000772  
 580 003626 012706 001100  
 581 003632 004767 000226  
 582 003636 032737 040000 177570  
 003644 001342

584

```

.SBTL  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
$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      ,SENDMG      ;;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      ,SENULL      ;;TYPE A NULL CHARACTER
003714      013700      000042      $GET42: MOV      @#42,R0  ;;GET MONITOR ADDRESS
003720      001405      BEG      $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: .ASCIIZ      <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$,@#MMVEC    ;SET UP FOR MEMORY MANAGEMENT ABORTS
591 004002 012700 010502      MOV    #END,R0       ;FORM ADDRESS OF WHERE TO START WRITING THE DATA
592 004006 062700 120000      ADD    #120000,R0    ;USE KIPAR5 FOR ADDRESSING
593 004012 012737 000001 177572  MOV    #1,@#MMRO     ;TURN ON MEMORY MANAGEMENT
594 004020 010520 1$:      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,@#MMRO     ;CLEAR MEMORY MANAGEMENT ERRORS
601 004050 000002      RTI                ;RETURN TO CONTINUE WRITING
602 004052 005037 177572 3$:      CLR    @#MMRO       ;TURN OFF MEMORY MANAGEMENT
603 004056 062706 000004      ADD    #4,SP        ;CLEAN UP THE STACK
604 004062 000207      RTS    PC           ;RETURN TO CALLER
  
```

CHECK MEMORY FOR CORRECT DATA PATTERN

```

605 .SBTTL CHECK MEMORY FOR CORRECT DATA PATTERN
606
607 004064 004767 000500 CHECK: JSR PC,MAP ;SET UP MEMORY MANAGEMENT REGISTERS
608 004070 016704 000470 MOV LIMIT,R4 ;GET BANK COUNT
609 004074 016705 000466 MOV DATA,R5 ;GET DATA PATTERN
610 004100 012737 004220 000250 MOV #2$,@#MMVEC ;SET UP FOR MEMORY MANAGEMENT ABORTS
611 004106 012700 010502 MOV #END,R0 ;ADDRESS OF WHERE DATA PATTERN STARTS
612 004112 062700 120000 ADD #120000,R0 ;USE KIPAR5 FOR ADDRESSING
613 004116 012737 000001 177572 MOV #1,@#MMR0 ;TURN ON MEM. MANAGEMENT
614 004124 012001 1$: MOV (R0)+,R1 ;READ THE DATA
615 004126 020501 CMP R5,R1 ;IS IT GOOD?
616 004130 001775 BEQ 1$ ;BRANCH IF YES
617 004132 010067 003542 MOV R0,VIR.ADD
618 004136 162767 000002 003534 SUB #2,VIR.ADD
619 004144 042767 160000 003526 BIC #160000,VIR.ADD
620 004152 013767 172352 003522 MOV @#KIPAR5,PAR.OFF
621 004160 010567 003474 MOV R5,EXPDAT
622 004164 010167 003472 MOV R1,RECDAT
623 004170 012706 000500 MOV #STACK1,SP ;SETUP RESERVED STACK POINTER
;FOR ERROR PRINTOUT
004174 104000 ERROR
;ERROR
;-----
624 004176 007700 VIR.ADD ;PRINT VIRTUAL ADDRESS
625 004200 007702 PAR.OFF ;PRINT PDR OFFSET
626 004202 007660 EXPDAT ;PRINT EXPECTED DATA
627 004204 007662 RECDAT ;PRINT RECEIVED DATA
628 004206 000000 0
629 004210 000000 HALT
630 004212 010560 177776 MOV R5,-2(R0) ;WRITE GOOD DATA
631 004216 000742 BR 1$ ;GO READ THE NEXT WORD
632 004220 005304 2$: DEC R4 ;LAST BANK COMPARED?
633 004222 100411 BMI 3$ ;BRANCH IF YES
634 004224 012700 120000 MOV #120000,R0 ;RESET ADDRESSING REGISTER
635 004230 062737 000200 172352 ADD #200,@#KIPAR5 ;SIZE NEXT 4K BANK
636 004236 012737 000001 177572 MOV #1,@#MMR0 ;CLEAR MEMORY MANAGEMENT ERRORS
637 004244 000002 RTI ;CONTINUE TESTING
638 004246 005037 177572 3$: CLR @#MMR0 ;TURN OFF MEM. MANAGEMENT
639 004252 062706 000004 ADD #4,SP ;CLEAN UP THE STACK
640 004256 000207 RTS PC ;RETURN FOR CALL

```



```

        .SBTTL  PARITY ERROR HANDLER
641
642
643 004260 005327      PARERR: D'      (PC)+      ;FIRST TIME IN?
644 004262 000001      PARFLG:          1
645 004264 002002          ;
646 004266 000000      1$:          2$      ;BRANCH IF YES
647 004270 000776          ;NO--ANOTHER PARITY ERROR OCCURRED WHILE
648 004272 032767 000077 173442 2$:      BR      1$      ;PROCESSING THE FIRST ONE.
649 004300 001006          BIT      #77,HIADRS ;MSB'S OF PARITY ERROR ADDRESS=0?
650 004302 023727 177740 010502 4$:      BNE     4$      ;BRANCH IF NO
651 004310 101002          CMP      @#LOADRS,#END ;LSB'S ABOVE THE PROGRAM?
652 004312 000000          BHI      4$      ;BRANCH IF YES
653 004314 000776          3$:      HALT   ;PARITY ERROR IS IN THE PROGRAM
654 004316 000000          BR      3$
655 004320 005737 177744          4$:      HALT   ;DATA PARITY ERROR OCCURRED
656 004324 100405          TST      @#MEMERR ;DID AN ABORT OCCUR?
657 004326 162700 000002          BMI     5$      ;BRANCH IF YES
658 004332 012705 000002          SUB      #2,R0 ;BACKUP BY ONE WORD
659 004336 074500          MOV      #BIT01,R5
660 C04340 010320          XOR      R5,R0
661 004342 013737 177744 177744 5$:      MOV      R3,(R0)+ ;REWRITE THE BAD DATA
662 004350 012767 000001 177704      MOV      @#MEMERR,@#MEMERR ;CLEAR ERROR INDICATORS
663 004356 000002          MOV      #1,PARFLG ;INITIALIZE PARITY ERROR FLAG
        RTI          ;CONTINUE TESTING
    
```

```

664      .SBTTL POWER FAIL ROUTINE
665      :*****
666 004360 012767 177777 000174 POWDWN: MOV    #-1,FLAG      ;FIRST INSTRUCTION FLAG
667 004366 005067 000170      CLR    FLAG          ;NOW CLEAR IT
668 004372 012777 004522 000150      MOV    #ILLUP,@UVEC   ;IF TOO FAST
669 004400 011667 000142      MOV    (SP),ERRAD     ;SET THE ERROR ADDRESS
670 004404 022706 000400      CMP    #400,SP        ;YELLOW OR RED?
671 004410 100402      BMI    1$            ;NO
672 004412 012706 001100      MOV    #STACK,SP     ;SET EMERGENCY STACK
673 004416 010046      1$:  MOV    R0,-(R6)      ;PUT
674 004420 010146      MOV    R1,-(R6)      ;THE
675 004422 010246      MOV    R2,-(R6)      ;REGISTERS
676 004424 010346      MOV    R3,-(R6)      ;ON
677 004426 010446      MOV    R4,-(R6)      ;THE
678 004430 010546      MOV    R5,-(R6)      ;STACK
679 004432 010667 000122      MOV    SP,SAV6       ;SAVE THE STACK POINTER
680 004436 004737 172240      JSR    PC,@#SIPARO   ;SET TIME FACTOR
681 004442 012777 004454 000100      MOV    #POWUP,@UVEC  ;RESET THE UP VECTOR
682 004450 000000      HALT                ;WAIT FOR POWER DOWN
683 004452 000240      NOP
684
685 004454 016706 000100      POWUP: MOV    SAV6,SP   ;RESET SP
686 004460 012777 004534 000066      MOV    #ILLDWN,@DVEC ;SET TOO FAST DOWN VECTOR
687 004466 004767 000200      JSR    PC,SETPAR     ;GO PUT TIME FACTOR SUBROUTINE IN SIPARO-SIPAR3
688 004472 004737 172240      JSR    PC,@#SIPARO   ;SET TIME FACTOR
689 004476 012605      MOV    (R6)+,R5      ;TAKE
690 004500 012604      MOV    (R6)+,R4      ;THE
691 004502 012603      MOV    (R6)+,R3      ;REGISTERS
692 004504 012602      MOV    (R6)+,R2      ;FROM
693 004506 012601      MOV    (R6)+,R1      ;THE
694 004510 012600      MOV    (R6)+,R0      ;STACK
695 004512 012777 004360 000034      MOV    #POWDWN,@DVEC ;RESET THE DOWN VECTOR
696 004520 000113      JMP    (R3)          ;JUMP INDIRECT TO R3
697
698 004522 104401      ILLUP: TYPE                ;POWER UP BEFORE POWER DOWN COMPLETE
699 004524 010037      UPF
700 004526 000000      HALT
701 004530 000167 173444      JMP    200
702 004534 104401      ILLDWN: TYPE              ;POWER DOWN BEFORE UP COMPLETE
703 004536 010107      DOWN
704 004540 000000      HALT
705 004542 000167 173432      JMP    200
706
707
708 004546 000000      ERRAD: .WORD    0      ;RETURN ADDRESS FROM POWER FAIL
709 004550 000024 000026      UVEC:  .WORD    24,26  ;UP ADDRESS PAIR
710 004554 000024 000026      DVEC:  .WORD    24,26  ;DOWN ADDRESS PAIR
711 004560 000000      SAV6:  .WORD    0      ;SOME PLACE TO PUT THE SP
712 004562 000000      FLAG:  .WORD    0      ;1 INSTRUCTION DOWN FLAG
713 004564 000000      LIMIT: .WORD    0      ;TOP OF MEMORY
714 004566 052525      DATA: .WORD    52525  ;WHAT IS TO BE WRITTEN INTO MEMORY

```

```
715          .SBTTL  SETUP MEMORY MANAGMENT REGISTERS
716
717 004570 012737 000000 172340 MAP:  MOV  #0,@#KIPAR0  ;SETUP PAR0 FOR 1ST 4K
718 004576 012737 077406 172300      MOV  #77406,@#KIPDR0 ;4K, R/W, EXPAND UP
719 004604 012737 000000 172352      MOV  #0,@#KIPAR5  ;SET UP PAR5 FOR 1ST 4K
720 004612 012737 077406 172312      MOV  #77406,@#KIPDR5 ;4K, R/W, ED=UP
721 004620 012737 000200 172354      MOV  #200,@#KIPAR6  ;SET UP PAR6 FOR 2ND 4K
722 004626 012737 000000 172314      MOV  #0,@#KIPDR6  ;ABORT ALL REFERENCES
723 004634 012737 177600 172356      MOV  #177600,@#KIPAR7;SET UP PAR7 FOR I/O PAGE
724 004642 012737 077406 172316      MOV  #77406,@#KIPDR7 ;4K, R/W, ED=UP
725 004650 012737 000020 172516      MOV  #BIT04,@#MMR3  ;SET UP FOR 22-BIT MAPPING
726 004656 012737 004666 000250      MOV  #MMERR,@#MMVEC ;SET UP MEMORY MANAGEMENT VECTOR
727 004664 000207          RTS      PC ;RETURN FROM CALL
728 004666 000000          MMERR: HALT ;MEMORY MANAGEMENT ERROR
729 004670 000776          BR      MMERR
```

```

737)
731 004672 012737 012700 172240 .SBTTL SUBROUTINE TO LOAD SIPARO-SIPAR3 WITH A SUBROUTINE
732 004700 012737 001777 172242 SETPAR: MOV #12700,@#SIPARO ;LOAD PARO & 1 WITH MOV #1777,R0
733 004706 012737 077001 172244 MOV #1777,@#SIPAR1
734 004714 012737 000207 172246 MOV #77001,@#SIPAR2 ;LOAD PAR2 WITH SOB RO,.
735 004722 000207 MOV #207,@#SIPAR3 ;LOAD PAR3 WITH RTS PC
736 RTS PC
737 .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
$SIZE: MOV R0,-(SP) ;;SAVE R0 ON THE STACK
MOV R1,-(SP) ;;SAVE R1 ON THE STACK
MOV R2,-(SP) ;;SAVE R2 ON THE STACK
MOV R3,-(SP) ;;SAVE R3 ON THE STACK
MOV R4,-(SP) ;;SAVE R4 ON THE STACK
MOV @#114,-(SP) ;;SAVE MEMORY ERROR VECTOR PS & PC
MOV @#116,-(SP)
MOV #116,@#114 ;;IGNORE PARITY ERRORS WHILE SIZING
MOV #RTI,@#116
MOV @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
MOV @#ERRVEC+2,-(SP)
MOV SP,R0 ;;SAVE THE STACK POINTER
;;SET THE ERRVEC PS TO THE PRESENT PS
TRAP ;;PUSH OLD PSW AND PC ON STACK
MOV (SP)+,@#ERRVEC+2 ;;SAVE THE PSW IN @#ERRVEC+2
MOV #3776,R1 ;;SETUP ADDRESS
TSTB (PC)+ ;;USE MEMORY MANAGEMENT?
$KT11: .WORD 200 ;;SET TO USE MEMORY MANAGEMENT
BPL $SCORE ;;BR IF NO
MOV # $KTINEX,@#ERRVEC ;;SET FOR TIMEOUT
TST @#SRO ;;KT11 ARE YOU THERE?
BIS #100000,$KT11 ;;YES--SET KT11 KEY
MOV #100$,@#ERRVEC ;;SET FOR TIMEOUT
TST @#170200 ;;UNIBUS MAP ARE YOU THERE?
MOV #176200,@#$STOP ;;YES-SET COMPARISON VALUE FOR 11/70
MOV #200,@#$MAP ;;TURN ON MAP INDICATOR
BR $MAPRG ;;GO SET UP MAP REGISTERS
MOV #6200,@#$STOP ;;COMPARISON VALUE FOR 18 BIT MAPPING
CMP (SP)+,(SP)+ ;;CLEAN OFF STACK
CLR @#$MAP ;;MAKE SURE MAP INDICATOR TURNED OFF
BR $NOMAP ;;
$MAP: .WORD 0 ;;=200 IF MAP PRESENT
$STOP: .WORD 0 ;;FILLED WITH APPROPRIATE COMPARISON VALUE
$MAPRG: MOV #37,R3 ;;SET UP COUNTER
MOV #170200,R2 ;;START WITH MAPLO
100$: CLR (R2)+ ;;LOAD ALL MAP REGISTERS

```

BK001

BK001

BK001

BK001

BK001

BK001

BK001

BK001

BK001

BK001

BK001

BK001

BK001

BK001

```

005120 012722 000074      MOV      #74,(R2)+      ;;WITH THE VALUE 1700000      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              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      # $MMOUT,@#114 ;;SET UP MEMORY ERROR VECTOR      BK001
005232 013737 000006 000116      MOV      @#ERRVEC+2,@#116 ;;LOCK OUT INTERRUPTS
005240 005737 143776      4$:  TST      @#143776   ;;TRAP ON NON-EX-MEM      BK001
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 $SCORE: 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              MCV     (SP)+,R2   ;;RESTORE R2

```

CHKACCO 11/44 POWER FAIL  
ROUTINE TO SIZE MEMORY

MACRO M1113 06-APR-81 14:07 PAGE 25-2 <sup>M 3</sup>

SEQUENCE 38

005426 012601  
005430 012600  
005432 000207  
005434 000000  
005436 000000

MOV (SP)+,R1        ::RESTORE R1  
MOV (SP)+,R0        ::RESTORE R0  
RTS PC  
\$LSTAD: .WORD 0        ::CONTAINS THE LAST ADDRESS  
\$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     $CMODE          ;;TYPE THIS DIGIT?
005566 100016      BPL      7$              ;;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

```

```

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$:   MOV   (SP)+,R5      ;;RESTORE R5
005642 012604          MOV   (SP)+,R4      ;;RESTORE R4
005644 012603          MOV   (S?)+,R3     ;;RESTORE R3
005646 016666 000002 000004      MOV   2(SP),4(SP)   ;;SET THE STACK FOR RETURNING
005654 012616          MOV   (SP)+,(SP)
005656 000002          RTI                    ;;RETURN
005660          000          8$:   .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
$TYPDS:
005666      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
005670      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
005672      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
005674      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
005676      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
005700      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
005704      MOV      20(SP),R5     ;;GET THE INPUT NUMBER
005710      BPL      1$           ;;BR IF INPUT IS POS.
005712      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
005714      MOVVB   #'-,1(SP)     ;;MAKE THE ASCII NUMBER NEG.
005722      CLR      R0           ;;ZERO THE CONSTANTS INDEX
005724      MOV      #$DBLK,R3     ;;SETUP THE OUTPUT POINTER
005730      MOVVB   #' ,(R3)+     ;;SET THE FIRST CHARACTER TO A BLANK
005734      CLR      R2           ;;CLEAR THE BCD NUMBER
005736      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
005742      SUB     R1,R5         ;;FORM THIS BCD DIGIT
005744      BLT     4$           ;;BR IF DONE
005746      INC     R2           ;;INCREASE THE BCD DIGIT BY 1
005750      BR      3$
005752      ADD     R1,R5         ;;ADD BACK THE CONSTANT
005754      TST     R2           ;;CHECK IF BCD DIGIT=0
005756      BNE     5$           ;;FALL THROUGH IF 0
005760      TSTB   (SP)         ;;STILL DOING LEADING 0'S?
005762      BMI     7$           ;;BR IF YES
005764      ASLB   (SP)         ;;MSD?
005766      BCC     6$           ;;BR IF NO
005770      MOVVB   1(SP),-1(R3)  ;;YES--SET THE SIGN
005776      BIS     #'0,R2       ;;MAKE THE BCD DIGIT ASCII
006002      BIS     #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
006006      MOVVB   R2,(R3)+     ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
006010      TST     (R0)+        ;;JUST INCREMENTING
006012      CMP     R0,#10       ;;CHECK THE TABLE INDEX
006016      BLT     2$           ;;GO DO THE NEXT DIGIT
006020      BGT     8$           ;;GO TO EXIT
006022      MOV     R5,R2        ;;GET THE LSD
006024      BR      6$           ;;GO CHANGE TO ASCII
006026      TSTB   (SP)+        ;;WAS THE LSD THE FIRST NON-ZERO?
006030      BPL     9$           ;;BR IF NO
006032      MOVVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
006040      CLRB   (R3)         ;;SET THE TERMINATOR
006042      MOV     (SP)+,R5     ;;POP STACK INTO R5
006044      MOV     (SP)+,R3     ;;POP STACK INTO R3
006046      MOV     (SP)+,R2     ;;POP STACK INTO R2
006050      MOV     (SP)+,R1     ;;POP STACK INTO R1
006052      MOV     (SP)+,R0     ;;POP STACK INTO R0
006054      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
```

247

```

.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: $FILLC 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$: MOV RO,-(SP) ;;SAVE RO
006126 017600 000002 MOV @2(SP),RO ;;GET ADDRESS OF ASCIZ STRING
006132 122767 000001 172774 CMPB #APTENV,$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 RO,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$: MOVB (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
006176 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
006200 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
006202 012600 60$: MOV (SP)+,RO ;;RESTORE RO
006204 062716 000002 3$: 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	C00007	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 EXTRANEIOUS 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 KDB 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>	::CARRIAGE RETURN
006474	012	000			\$LF:	.ASCII	<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)

```

```

747          .SBTTL  ERROR PRINT ROUTINE
748          ERROR  =104000
749 006546   000000
750 006550   010067   000026
751 006554   010167   000024
752 006560   010267   000022
753 006564   010367   000020
754 006570   010467   000016
755 006574   010567   000014
756 006600   000406
757 006602   000000
758 006604   000000
759 006606   000000
760 006610   000000
761 006612   000000
762 006614   000000
763
764 006616   011601
765 006620   162701   000002
766 006624   010167   177716
767 006630   104401   006636
   006634   000411

          .WORD 0
          MOV   R0,SAVR0      ;SAVE R0 THRU R5
          MOV   R1,SAVR1
          MOV   R2,SAVR2
          MOV   R3,SAVR3
          MOV   R4,SAVR4
          MOV   R5,SAVR5
          BR    TITL

SAVR0:   .WORD
SAVR1:   .WORD
SAVR2:   .WORD
SAVR3:   .WORD
SAVR4:   .WORD
SAVR5:   .WORD

TITL:    MOV   (SP),R1      ;R1 CONTAINS ADDRESS FOLLOWING ERRPC ADDRESS
          SUB   #2,R1
          MOV   R1,ERRPC    ;GET ADDRESS OF ERROR MESSAGE:ERRPC
          TYPE  ,65$        ;;TYPE ASCIZ STRING
          BR    ,64$        ;;GET OVER THE ASCIZ
          .ASCIZ <CRLF><CRLF>/TESTNO  ERRPC/
          .ASCIZ <CRLF><CRLF>/TESTNO  ERRPC/
          .ASCIZ <CRLF><CRLF>/TESTNO  ERRPC/

          TST   (R1)+      ;R1 POINTS TO NEXT ARGUEMENT

          MOV   #PRTABL,R2  ;ADDRESS OF START OF PRINT TABLE LIST
          MOV   #PRTITL,R3  ;ADDRESS OF START OF TITLES
          TST   (R1)        ;END OF ARGUEMENTS?
          BEQ   DAT        ;YES,GO PRINT DATA
          TST   (R3)+      ;INDEX THRU TITLES
          CMP   (R1),(R2)+  ;SEARCH PRINT TABLE LIST FOR TITLE
          BNE   2$        ;NO; CHECK NEXT LOCATION IN LIST

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

DAT:     TYPE  ,65$        ;;TYPE ASCIZ STRING
          BR    ,64$        ;;GET OVER THE ASCIZ
          .ASCIZ <CRLF>
          .ASCIZ <CRLF>

          MOV   $TESTN,-(SP) ;;SAVE $TESTN FOR TYPEOUT
          TYPOS ;GO TYPE--OCTAL ASCII
          .BYTE 6          ;;TYPE 6 DIGIT(S)
          .BYTE 0          ;;SUPPRESS LEADING ZEROS
          TYPE  ,67$        ;;TYPE ASCIZ STRING
          BR    ,66$        ;;GET OVER THE ASCIZ
          .ASCIZ / /
          .ASCIZ / /

          MOV   ERRPC,-(SP) ;;SAVE ERRPC FOR TYPEOUT
          TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
          TYPE  , $ENULL

          MOV   (SP),R1      ;R1 CONTAINS ADDRESS FOLLOWING ERRORPC ADDRESS
          SUB   #2,R1
          TST   (R1)+      ;R1 POINTS TO NEXT ARGUEMENT
  
```

```

790 006764 012702 007634      MOV      #PRTABL,R2      ;ADDRESS OF START OF PRINT TABLE LIST
791 006770 012703 007470      MOV      #PRDATA,R3     ;ADDRESS OF START OF PRINT DATA
792 006774 005711              TST      (R1)           ;END OF ARGUMENTS?
793 006776 001421              BEQ      FIN            ;YES
794 007000 005723      2$:    TST      (R3)+         ;INDEX THRU DATA PRINTS
795 007002 021122      CMP      (R1),(R2)+     ;SEARCH PRINT TABLE LIST FOR TITLE
796 007004 001375      BNE      2$            ;NO; CHECK NEXT LOCATION IN LIST
797 007006 104401 007014      TYPE    ,69$          ;;TYPE ASCIZ STRING
      BR     68$          ;;GET OVER THE ASCIZ
      ;;69$: .ASCIZ / /
      68$:
      007020
798 007020 004753              JSR      PC,@-(R3)      ;
799 007022 104401 003740      TYPE    , $NULL
800 007026 104401 007034      TYPE    ,71$          ;;TYPE ASCIZ STRING
      BR     70$          ;;GET OVER THE ASCIZ
      ;;71$: .ASCIZ / /
      70$:
801 007040 000750              BR      3$
802
803 007042 005721      FIN:    TST      (R1)+   ;R1 NOW CONTAINS RETURN ADDRESS
804 007044 010116      MOV      R1,(SP)      ;SETUP RETURN ADDRESS IN STACK
805 007046 104401 007054      TYPE    ,65$          ;;TYPE ASCIZ STRING
      BR     64$          ;;GET OVER THE ASCIZ
      ;;65$: .ASCIZ <CRLF>
      64$:
      007056
806
807 007056 016700 177520      MOV      SAVR0,R0      ;RESTORE REGISTERS
808 007062 016701 177516      MOV      SAVR1,R1
809 007066 016702 177514      MOV      SAVR2,R2
810 007072 016703 177512      MOV      SAVR3,R3
811 007076 016704 177510      MOV      SAVR4,R4
812 007102 016705 177506      MOV      SAVR5,R5
813
814 007106 000002      RTI      ;RETURN
  
```

```

815 .SBTTL SUBROUTINES TO PRINT DATA HEADERS
816 007110 007134 007162 007210 PRITTL: .WORD 1$,2$,3$,4$,5$,6$,7$,10$,11$,12$
817 007134 104401 007142 1$: TYPE ,65$ ::TYPE ASCIZ STRING
      007140 000407 BR 64$ ::GET OVER THE ASCIZ
      ::65$: .ASCIZ / EXPDAT /
      64$:
818 007160
819 007160 000207 2$: RTS PC
      007162 104401 007170 TYPE ,67$ ::TYPE ASCIZ STRING
      007166 000407 BR 66$ ::GET OVER THE ASCIZ
      ::67$: .ASCIZ / RECDAT /
      66$:
820 007206
821 007210 000207 3$: RTS PC
      007210 104401 007216 TYPE ,69$ ::TYPE ASCIZ STRING
      007214 000407 BR 68$ ::GET OVER THE ASCIZ
      ::69$: .ASCIZ / EXP.ADD /
      68$:
822 007234
823 007236 000207 4$: RTS PC
      007236 104401 007244 TYPE ,71$ ::TYPE ASCIZ STRING
      007242 000407 BR 70$ ::GET OVER THE ASCIZ
      ::71$: .ASCIZ / REC.ADD /
      70$:
824 007262
825 007264 000207 5$: RTS PC
      007264 104401 007272 TYPE ,73$ ::TYPE ASCIZ STRING
      007270 000407 BR 72$ ::GET OVER THE ASCIZ
      ::73$: .ASCIZ / EXP.SP1 /
      72$:
826 007310
827 007312 000207 6$: RTS PC
      007312 104401 007320 TYPE ,75$ ::TYPE ASCIZ STRING
      007316 000407 BR 74$ ::GET OVER THE ASCIZ
      ::75$: .ASCIZ / REC.SP1 /
      74$:
828 007336
829 007340 000207 7$: RTS PC
      007340 104401 007346 TYPE ,77$ ::TYPE ASCIZ STRING
      007344 000407 BR 76$ ::GET OVER THE ASCIZ
      ::77$: .ASCIZ / EXP.PSW /
      76$:
830 007364
831 007366 000207 10$: RTS PC
      007366 104401 007374 TYPE ,79$ ::TYPE ASCIZ STRING
      007372 000407 BR 78$ ::GET OVER THE ASCIZ
      ::79$: .ASCIZ / REC.PSW /
      78$:
832 007412
833 007414 000207 11$: RTS PC
      007414 104401 007422 TYPE ,81$ ::TYPE ASCIZ STRING
      007420 000407 BR 80$ ::GET OVER THE ASCIZ
      ::81$: .ASCIZ / VIR.ADD /
      80$:
834 007440
835 007442 000207 12$: RTS PC
  
```



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

```

837 .SBTTL SUBROUTINES TO PRINT DATA
838 007470 007514 007524 007534 PRDATA: .WORD 1$,2$,3$,4$,5$,6$,7$,10$,11$,12$
839 007514 1$:
      007514 016746 000140 MOV EXPDAT,-(SP) ;;SAVE EXPDAT FOR TYPEOUT
      007520 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
840 007522 000207 RTS PC
841 007524 2$:
      007524 016746 000132 MOV RECDAT,-(SP) ;;SAVE RECDAT FOR TYPEOUT
      007530 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
842 007532 000207 RTS PC
843 007534 3$:
      007534 016746 000124 MOV EXP.ADD,-(SP) ;;SAVE EXP.ADD FOR TYPEOUT
      007540 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
844 007542 000207 RTS PC
845 007544 4$:
      007544 016746 000116 MOV REC.ADD,-(SP) ;;SAVE REC.ADD FOR TYPEOUT
      007550 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
846 007552 000207 RTS PC
847 007554 5$:
      007554 016746 000110 MOV EXP.SP1,-(SP) ;;SAVE EXP.SP1 FOR TYPEOUT
      007560 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
848 007562 000207 RTS PC
849 007564 6$:
      007564 016746 000102 MOV REC.SP1,-(SP) ;;SAVE REC.SP1 FOR TYPEOUT
      007570 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
850 007572 000207 RTS PC
851 007574 7$:
      007574 016746 000074 MOV EXP.PSW,-(SP) ;;SAVE EXP.PSW FOR TYPEOUT
      007600 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
852 007602 000207 RTS PC
853 007604 10$:
      007604 016746 000066 MOV REC.PSW,-(SP) ;;SAVE REC.PSW FOR TYPEOUT
      007610 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
854 007612 000207 RTS PC
855 007614 11$:
      007614 016746 000060 MOV VIR.ADD,-(SP) ;;SAVE VIR.ADD FOR TYPEOUT
      007620 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
856 007622 000207 RTS PC
857 007624 12$:
      007624 016746 000052 MOV PAR.OFF,-(SP) ;;SAVE PAR.OFF FOR TYPEOUT
      007630 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
858 007632 000207 RTS PC

```

```

859                                     .SBTTL  TABLES AND ASCII MESSAGES
860 007634 007660 007662 007664 PRTABL: .WORD  EXPDAT,RECDAT,EXP.ADD,REC.ADD,EXP.SP1
861 007646 007672 007674 007676       .WORD  REC.SP1,EXP.PSW,REC.PSW,VIR.ADD,PAR.OFF
862 007660 000000       EXPDAT: .WORD  0
863 007662 000000       RECDAT: .WORD  0
864 007664 000000       EXP.ADD: .WORD  0
865 007666 000000       REC.ADD: .WORD  0
866 007670 000000       EXP.SP1: .WORD  0
867 007672 000000       REC.SP1: .WORD  0
868 007674 000000       EXP.PSW: .WORD  0
869 007676 000000       REC.PSW: .WORD  0
870 007700 000000       VIR.ADD: .WORD  0
871 007702 000000       PAR.OFF: .WORD  0
872 007704      015      012      103 PNAME: .ASCIIZ <15><12>+CKKACCO 11/44 POWER FAIL+
873 007737      015      012      103 TITLE: .ASCIIZ <15><12>+CKKACCO 11/44 POWER FAIL+<15><12>
874 007774      015      012      102 BOOT: .ASCIIZ <15><12>+BOOT ENABLE SWITCH MUST BE OFF+<15><12>
875 010037      015      012      120 UPF: .ASCIIZ <15><12>+POWER UP BEFORE POWER DOWN COMPLETE+<15><12>
876 010107      015      012      120 DOWN: .ASCIIZ <15><12>+POWER DOWN BEFORE UP COMPLETE+<15><12>
877 010151      055      113      040 SUPMSG: .ASCII  ?-K WORDS OF MEMORY EXIST?<CRLF>
878 010202      122      101      116       .ASCII  ?RANDOM CHARACTER(S) (INCLUDING FORM FEED) ?
879 010254      115      101      131       .ASCIIZ  ?MAY PRINT ON POWER CYCLING.?<CRLF>
880 010311      115      101      116 USRPFL: .ASCIIZ  ?MANUALLY INTERRUPT THE POWER AFTER EACH TEST #?<CRLF>
881 010371      116      117      040 HDWPFL: .ASCII  ?NO MANUAL INTERVENTION NECESSARY - ?
882 010434      101      125      124       .ASCIIZ  ?AUTO POWER FAIL TEST HARDWARE EXISTS?<CRLF>
883                                     .EVEN
884 010502 000000       END: 0
885      000001       .END

```

ABASE = 000000	BIT03 = 000010	KDPARO= 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	MMRO = 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
ADEVVM = 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	CACHE= 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	PRO = 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 =X000006	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	MAPHO = 170202	MAPL17= 170274	REC.SP 007672
APTSIZ= 000200	EXPDAT 007660	MAPHO0= 170202	MAPL2 = 170210	RESVEC= 000010
APTSPO= 000100	EXP.AD 007664	MAPHO1= 170206	MAPL20= 170300	R10 =X000000
ASWREG= 000000	EXP.PS 007674	MAPHO2= 170212	MAPL21= 170304	R11 =X000001
ATESTN= 000000	EXP.SP 007670	MAPHO3= 170216	MAPL22= 170310	R12 =X000002
AUNIT = 000000	FIN 007042	MAPHO4= 170222	MAPL23= 170314	R13 =X000003
AUSWR = 000000	FLAG 004562	MAPHO5= 170226	MAPL24= 170320	R14 =X000004
AVECT1= 000000	HDWPFL 010371	MAPHO6= 170232	MAPL25= 170324	R15 =X000005
AVECT2= C.0000	HIADRS= 177742	MAPHO7= 170236	MAPL26= 170330	R6 =X000006
BEGIN 001424	HITMIS= 177752	MAPH1 = 170206	MAPL27= 170334	R7 =X000007
BIT0 = 000001	HT = 000011	MAPH10= 170242	MAPL3 = 170214	SAVRO 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

SAVR4	006612	SUPMSG	010151	TYPDS	= 104405	\$ATY3	C01156	\$MYP1	001145
SAVR5	006614	SUPSTK	= 000700	TYPE	= 104401	\$ATY4	001166	\$NOMAP	005126
SAV6	004560	SWR	= 177570	TYPOC	= 104402	\$CHARC	006452	\$NULL	006466
SCOPE	= 000004	SWREG	000176	TYPON	= 104404	\$CORE	005326	\$NWTST	= 000001
SDPAR0	= 172260	SW0	= 000001	TYPOS	= 104403	\$CPUOP	001142	\$OCNT	005662
SDPAR1	= 172262	SW00	= 000001	UDPAR0	= 177660	\$CRLF	006473	\$OMODE	005664
SDPAR2	= 172264	SW01	= 000002	UDPAR1	= 177662	\$CROUT	005356	\$PASS	001122
SDPAR3	= 172266	SW02	= 000004	UDPAR2	= 177664	\$DBLK	006102	\$PASTM	001106
SDPAR4	= 172270	SW03	= 000010	UDPAR3	= 177666	\$DEVCT	001124	\$QUES	006472
SDPAR5	= 172272	SW04	= 000020	UDPAR4	= 177670	\$DOAGN	003734	\$RTNAD	003736
SDPAR6	= 172274	SW05	= 000040	UDPAR5	= 177672	\$DTBL	006072	\$SAVPC	= 000204
SDPAR7	= 172276	SW06	= 000100	UDPAR6	= 177674	\$ENDAD	003724	\$SETUP	= 000024
SDPDR0	= 172220	SW07	= 000200	UDPAR7	= 177676	\$ENDCT	003672	\$SIZE	004724
SDPDR1	= 172222	SW08	= 000400	UDPDR0	= 177620	\$ENDMG	003743	\$SIZEX	005362
SDPDR2	= 172224	SW09	= 001000	UDPDR1	= 177622	\$ENULL	003740	\$STOP	005104
SDPDR3	= 172226	SW1	= 000002	UDPDR2	= 177624	\$ENV	001134	\$STUP	= 177777
SDPDR4	= 172230	SW10	= 002000	UDPDR3	= 177626	\$ENVM	001135	\$SVPC	= 000204
SDPDR5	= 172232	SW11	= 004000	UDPDR4	= 177630	\$EOP	003646	\$SWR	= 040000
SDPDR6	= 172234	SW12	= 010000	UDPDR5	= 177632	\$EOPCT	003664	\$SWREG	001136
SDPDR7	= 172236	SW13	= 020000	UDPDR6	= 177634	\$ERROR	006550	\$TESTN	001120
SETPAR	004672	SW14	= 040000	UDPDR7	= 177636	\$ETABL	001134	\$TKB	006460
SIPAR0	= 172240	SW15	= 100000	UIPAR0	= 177640	\$ETEND	001150	\$TKS	006456
SIPAR1	= 172242	SW2	= 000004	UIPAR1	= 177642	\$FATAL	001116	\$TN	= 000012
SIPAR2	= 172244	SW3	= 000010	UIPAR2	= 177644	\$FFLG	001414	\$TPB	006464
SIPAR3	= 172246	SW4	= 000020	UIPAR3	= 177646	\$FILLC	006470	\$TPFLG	006471
SIPAR4	= 172250	SW5	= 000040	UIPAR4	= 177650	\$FILLS	006467	\$TPS	006462
SIPAR5	= 172252	SW6	= 000100	UIPAR5	= 177652	\$GET42	003714	\$TRAP	006476
SIPAR6	= 172254	SW7	= 000200	UIPAR6	= 177654	\$HD	= 000000	\$TRAP2	006520
SIPAR7	= 172256	SW8	= 000400	UIPAR7	= 177656	\$HIBTS	001100	\$TRP	= 000006
SIPDR0	= 172200	SW9	= 001000	UIPDR0	= 177600	\$KTNEX	005320	\$TRPAD	006532
SIPDR1	= 172202	SYSTID	= 177764	UIPDR1	= 177602	\$KTOUT	005256	\$STSM	001104
SIPDR2	= 172204	TBITVE	= 000014	UIPDR2	= 177604	\$KT11	005010	\$TYPDS	005666
SIPDR3	= 172206	TITL	006616	UIPDR3	= 177606	\$LF	006474	\$TYPE	006112
SIPDR4	= 172210	TITLE	007737	UIPDR4	= 177610	\$LFLG	001413	\$TYPEC	006324
SIPDR5	= 172212	TKVEC	= 000060	UIPDR5	= 177612	\$LSTAD	005434	\$TYPEX	006454
SIPDR6	= 172214	TPVEC	= 000064	UIPDR6	= 177614	\$LSTBK	005436	\$TYPOC	005464
SIPDR7	= 172216	TRAPVE	000034	UIPDR7	= 177616	\$MADR1	001146	\$TYPON	005500
SIZEH1	= 177762	TRTVEC	= 000014	UPF	010037	\$MAIL	001114	\$TYPOS	005440
SIZELO	= 177760	TST1	002036	USESTK	= 000600	\$MAMS1	001144	\$UNIT	001126
SRO	= 177572	TST10	003420	USP	= 000006	\$MAP	005102	\$UNITM	001110
SR1	= 177574	TST11	003552	USRPF1	010311	\$MAPRG	005106	\$USWR	001140
SR2	= 177576	TST2	002254	UVEC	004550	\$MBADR	001102	\$XOFF	= 000023
SR3	= 172516	TST3	002504	VIR.AD	007700	\$MFLG	001412	\$XON	= 000021
SSP	= 000006	TST4	002764	XYZ	001420	\$MMOUT	005300	\$SET4	= 000000
STACK	= 001100	TST5	003070	\$APTHD	001100	\$MSGAD	001130	\$OFILL	005663
STACK1	= 000500	TST6	003200	\$ATYC	001174	\$MSGLG	001132	\$.SX	= 001100
STKMT	= 177774	TST7	003320	\$ATY1	001150	\$MSGTY	001114		

. 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		REFERENCES	
SYMBOL	VALUE		
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
ADEVCM	= 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	
APTC SU	= 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		REFERENCES									
SYMBOL	VALUE										
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									
CACHE	= 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	10-377	10-388	11-413	11-424	11-435	12-466	12-477		
		12-488	14-515	15-529	16-547	17-563	21-623	#30-748			
		13-503	30-784								
ERRPC	= 006546	#30-749	*30-766								
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

SYMBOL	CROSS REFERENCE VALUE	REFERENCES
EXP DAT	007660	*25-737 25-737 *25-737 *25-737 *25-737
EXP AD	007664	*21-621 21-626 32-839 33-860 #33-862
EXP PS	007674	*10-375 10-379 *11-433 11-437 *12-475 12-479 32-843 33-860 #33-864
EXP SP	007670	*10-386 10-390 *11-422 11-426 *12-486 12-490 32-851 33-861 #33-868
FIN	007042	*10-363 10-367 *11-411 11-415 *12-464 12-468 32-847 33-860 #33-866
FLAG	004562	30-793 #30-803
GNS	= *****	*15-523 *23-666 *23-667 #23-712
		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 30-767 30-781 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
HDW PFL	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	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	=	03760	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
MAP0	=	170202	#7-280
MAP00	=	170202	#7-280 7-280
MAP01	=	170206	#7-280 7-280
MAP02	=	170212	#7-280 7-280
MAP03	=	170216	#7-280 7-280
MAP04	=	170222	#7-280 7-280
MAP05	=	170226	#7-280 7-280
MAP06	=	170232	#7-280 7-280
MAP07	=	170236	#7-280 7-280
MAP1	=	170206	#7-280
MAP10	=	170242	#7-280
MAP11	=	170246	#7-280
MAP12	=	170252	#7-280
MAP13	=	170256	#7-280
MAP14	=	170262	#7-280
MAP15	=	170266	#7-280
MAP16	=	170272	#7-280
MAP17	=	170276	#7-280
MAP2	=	170212	#7-280
MAP20	=	170302	#7-280
MAP21	=	170306	#7-280
MAP22	=	170312	#7-280
MAP23	=	170316	#7-280
MAP24	=	170320	#7-280
MAP25	=	170326	#7-280
MAP26	=	170332	#7-280
MAP27	=	170336	#7-280
MAP3	=	170216	#7-280
MAP30	=	170342	#7-280
MAP31	=	170346	#7-280
MAP32	=	170352	#7-280
MAP33	=	170356	#7-280
MAP34	=	170362	#7-280
MAP35	=	170366	#7-280
MAP36	=	170372	#7-280
MAP37	=	170376	#7-280
MAP4	=	170222	#7-280
MAP5	=	170226	#7-280
MAP6	=	170232	#7-280
MAP7	=	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
MEMERR	004666	24-726 #24-728 24-729
MMRO	= 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
PFAL	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
PIRG	= 177772	#7-280

SYMBOL	CROSS REFERENCE	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 #33-860
PRTITL	=	007110	30-771 #31-816
PRO	=	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.SP	=	007676	*10-387 10-391 *11-423 11-427 12-487 12-491 32-853 33-861 #33-869
RESVEC	=	000010	*10-364 10-368 *11-412 11-416 *12-465 12-469 32-849 33-861 #33-867
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
SAVRO	=	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
SDPDRO	=	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	12-462
			12-472	12-483	13-501	14-512	14-517	16-541	16-550	17-561
			18-580	23-672						
STACK1	= 000500	#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	
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	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					
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					
JIPDR7	= 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				
\$SCORE	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								
\$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	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						
\$STUP	= 177777	#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							
\$TFSTN	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	*14-519	
		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	29-745
\$TRPAD	006532	29-745	#29-745							
\$STM	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
.SETUP	#7-238 #7-239 #7-288
.SURMI	#7-238 #7-247
.SURLO	#7-247
.SACT1	#7-239 #7-289
.SAPT8	#7-239 7-292
.SAPTH	#7-239 7-291
.SAPTY	#7-239 7-293
.SCATC	#7-238 #7-281
.SEOP	#7-238 19-584
.SPOWE	#7-238
.SSIZE	#7-239 25-737
.STRAP	#7-239 #29-745
.STYPD	#7-239 #27-741
.STYPE	#7-238 28-743
.STYPO	#7-238 26-739
.1170	#7-238 #7-280