

.REM %

MAINDEC-11-DEMJA-B PDP11/70 MEMORY TEST MACY11 27(732) 09-SEP-76 17:03 PAGE 1
DEMJOB.P11

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DEMJA-B-D
PRODUCT NAME:	PDP-11/70 MEMORY TEST
DATE CREATED:	MARCH 1976
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JIM LACEY

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 DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

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

COPYRIGHT (C) 1973 1976 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
- 3.0 LOADING & STARTING PROCEDURE
 - 3.1 ACT11 OPERATION
- 4.0 SWITCH SETTINGS
- 5.0 SUBROUTINE ABSTRACTS
 - 5.1 SCOPE
- 6.0 ERRORS
 - 6.1 PARITY ERROR
- 7.0 RESTRICTIONS
 - 7.1 STARTING RESTRICTION
 - 7.2 OPERATION RESTRICTION
- 8.0 MISCELLANEOUS
 - 8.1 STACK POINTER
 - 8.2 PASS COUNT
 - 8.3 ERROR COUNT
 - 8.4 DISPLAY REGISTER
 - 8.5 POWER FAIL
 - 8.6 EXECUTION TIME
- 9.0 PROGRAM DESCRIPTION

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
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

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
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150

1.0 ABSTRACT

PROGRAM DEMJA TESTS CONTIGUOUS MEMORY ADDRESS FROM 000000 TO 17757776. IT VERIFIES THAT EACH ADDRESS IS UNIQUE (AN ADDRESS TEST) AND THAT EACH MEMORY LOCATION CAN BE READ/WRITTEN RELIABLY (WORST CASE NOISE TESTS). THIS PROGRAM MAY BE USED TO ADJUST/MARGIN MEMORY.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11/70 FAMILY PROCESSOR WITH 32K MEMORY

2.2 STORAGE

PROGRAM STORAGE - THE PROGRAM USES MEMORY 0-17777

2.3 PRELIMINARY PROGRAMS

DEKBA THROUGH DEKBF

3.0 LOADING AND STARTING PROCEDURE

LOAD PROGRAM INTO MEMORY USING ABS LOADER

LOAD ADDRESS 200

SET SW12 IN DESIRED POSITION (SEE SEC 4.0)

PRESS START.

ASTERISK "*" WILL BE PRINTED AFTER EACH PASS.

"DEMJA DONE!" WILL BE PRINTED AFTER 6 PASSES.

PASS COUNT MAY BE MONITORED IN THE DISPLAY REGISTER.

NOTE: THIS PROGRAM SAVES THE LOADERS (BOOT AND ABS), TO RESTORE THE LOADERS, RESTART AT 162.

3.1 ACT11 OPERATION

IF THE PROGRAM IS RUN IN QUICK VERIFY MODE UNDER ACT11 THE PROGRAM IS DONE AFTER THE FIRST PASS..PAGE

4.0 SWITCH SETTINGS

SW15 = 1 OR UP.... HALT ON ERROR

NOTE: IF SW15=1 WHEN AN ERROR OCCURS THE PROGRAM WILL HALT, AND THE CORRECT DATA WILL NOT BE LOADED INTO THE FAILING ADDRESS. IF SW15 IS RAISED AFTER THE ERROR TYPEOUT BEGINS THE PROGRAM WILL HALT WHEN THE TYPEOUT COMPLETES, AND THE CORRECT DATA WILL BE LOADED INTO THE FAILING ADDRESS.

SW14 = 1 OR UP.... LOOP SUBTEST

SW13 = 1 OR UP..... INHIBIT ERROR TYPEOUT

SW12 = 1 OR UP....INHIBIT USE OF MEMORY MANAGEMENT

NOTE: INHIBITING THE USE OF MEMORY MANAGEMENT CAN

H01

MAINDEC-11-DEMJA-B PDP11/70 MEMORY TEST MACY11 27(732) 09-SEP-76 17:07 PAGE 4
DEMJOB.P11

151
152

BE DONE ONLY WHEN THE PROGRAM IS STARTED.
IF THE USE OF MEMORY MANAGEMENT IS INHIBITED THE LAST

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
180
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

ADDRESS AS TYPED BY THE PROGRAM WILL ONLY REFLECT THE AMOUNT OF MEMORY UP TO 28K (LAST ADDRESS = 160000).

SW11 = 1 OR UP..... INHIBIT SUBTEST ITERATION

SW10 = 1 OR UP..... RING BELL ON ERROR

SW9 = 1 OR UP..... DISPLAY ERROR COUNT IN DISPLAY REGISTER

SW9 = 0 OR DOWN... DISPLAY PASS COUNT IN DISPLAY REGISTER

SW8 = 1 OR UP..... HALT PROGRAM UNRELOCATED & RESTORE LOADERS.

5.0 SUBROUTINE ABSTRACTS

5.1 SCOPE

THE PROGRAM STORES IN R1 THE PC OF THE LAST TEST SUCCESSFULLY EXECUTED AND MAY BE USED AS AN AID IN DEBUGGING IF THE PROGRAM 'BOMBS' BECAUSE OF A HARDWARE FAILURE.

6.0 ERRORS

THESE TESTS PRINT OUT THE PC WHERE THE ERROR WAS DETECTED, THE FAILING ADDRESS, THE GOOD DATA, AND THE BAD DATA I.E.

PC=XXXXXX ADDRESS AAAAAA GOOD DATA GGGGGG BAD DATA BBBBBB

THE ADDRESS OF THE FAILING LOCATION IS THE TRUE 22 BIT PHYSICAL ADDRESS.

NOTE: WHEN TESTING MEMORY LOCATIONS 0-77776 THE PC TYPED WILL BE A MULTIPLE OF 100000 GREATER THAN REFLECTED IN THE PROGRAM LISTING

THE ADDRESS OF THE BAD DATA IS IN (R2) -2

THE GOOD DATA IN R0

THE BAD DATA IN R3

THE ADDRESS OF GOOD DATA IS IN R4 (RANDOM DATA TEST ONLY)

WHEN AN ERROR IS DETECTED WHEN EXERCISING THE MEMORY USING THE WORST CASE NOISE PATTERNS, THE USER SHOULD RESTART THE PROGRAM SELECTING PROGRAM #2 (SEE SEC 9.1 FOR DETAILS) SELECTING THE APPROPRIATE PARAMETERS. THE USER CAN USE THE PC AND ADDRESS OF THE FAILURE TO SELECT THE PROPER CORE BANK(S) AFFECTED AND ALSO THE SPECIFIC PATTERN. THIS ALLOWS MAXIMUM SCOPE CAPABILITIES.

6.1 PARITY ERROR

IF A PARITY ERROR IS DETECTED THE PROGRAM WILL TYPE:

PARITY ERROR

J01

MAINDEC-11-DEMJA-B PDP11/70 MEMORY TEST MACY11 27(732) 09-SEP-76 17:07 PAGE 6
DEMJOB.P11

209
210

PC=PPPPP MEMORY ADDRESS IS AAAAAAA
PARITY ERROR REG=EEEEEE ?????????? MARGIN

211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266

WHERE P P P P P IS THE CONTENTS OF THE PC WHEN THE PARITY ERROR OCCURRED, A A A A A A A IS THE ADDRESS OF THE WORD, E E E E E IS THE CONTENTS OF THE MEMORY ERROR REGISTER, AND ? ? ? ? ? ? ? ? ? ? IS THE MARGIN SETTING AT THE TIME OF THE PARITY ERROR.

AFTER REPORTING THE PARITY ERROR THE PROGRAM WILL START 'OVER.

7.0 RESTRICTIONS

7.1 STARTING RESTRICTION
PROGRAM MUST NOT BE RELOCATED WHEN RESTARTING

7.2 OPERATIONAL RESTRICTION
PROGRAM CHECKS CONTIGUOUS MEMORY IF A PARITY ERROR TRAP OCCURS WHEN THE PROGRAM IS RELOCATED PROGRAM ACTION IS UNDEFINED. IF PARITY MEMORY IS AVAILABLE OR SELECTED THE 3XOR9 TEST PATTERN IS FOR PARITY MEMORY ONLY. DO NOT POWER FAIL THE PROGRAM WHEN THE PROGRAM IS RUNNING RELOCATED.

8.0 MISCELLANEOUS

IF THE PROGRAM HALTS IN THE TRAP/INTERRUPT VECTOR AREA (0-1000), EXAMINE REGISTER 6 (THE STACK PTR). R6 CONTAINS THE ADDRESS WHERE THE PC OF THE INSTRUCTION THAT CAUSED THE TRAP ABORT IS STORED. SEE ALSO R1 (R1 SPECIFIES THE LAST TEST COMPLETED).

NOTE: THE PDP-11/70 WILL DISPLAY THE TRAP VECTOR ADDRESS+4 IN THE ADDRESS LIGHTS. THUS A TRAP TO 4 (BUS ERROR) WILL DISPLAY 10 IN THE ADDRESS LIGHTS.

8.1 STACK POINTER
THE STACK POINTER IS INITIALLY SET TO 520 AND IS RESET TO THIS VALUE AT THE START OF EACH SUBTEST.

8.2 PASS COUNT
SIX PASSES ARE REQUIRED FOR COMPLETION OF THIS PROGRAM; AT WHICH TIME AN "*" WILL BE PRINTED. THE PASS COUNT MAY BE OBSERVED BY TURNING THE SWITCH TO THE DISPLAY POSITION. (THE PASS COUNT IS ALSO STORED IN LOCATION 1000.) THE PASS COUNT SHOULD BE MONITORED IN THE EVENT THAT THE PROGRAM ENTERS AN UNDEFINED LOOP..BLANK 1

8.3 ERROR COUNT
EACH TIME AN ERROR OCCURS, THE ERROR COUNT IS INCREMENTED. THE ERROR COUNT CAN BE OBSERVED BY TURNING THE SWITCH TO THE DISPLAY POSITION AND SETTING SWITCH 9. (THE ERROR COUNT IS ALSO STORED IN LOCATION 1002.) THE PROGRAM WILL COUNT 17777(8) ERRORS; THE ERROR COUNT IS NOT INCREMENTED PAST THIS VALUE..BLANK 1

8.4 DISPLAY REGISTER
EITHER THE PASS COUNT OR THE ERROR COUNT IS DISPLAYED IN THE DISPLAY REGISTER. THE COUNT TO BE DISPLAYED IS CONTROLLED BY THE SETTING OF SWITCH 9..BLANK 1

LO1

MAINDEC-11-DEMJA-B PDP11/70 MEMORY TEST MACY11 27(732) 09-SEP-76 17:07 PAGE 8
DEMJOB.P11

267
268

8.5 POWER FAIL

269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324

THE PROGRAM MAY BE POWER FAILED WHEN RUNNING. WHEN THE POWER RETURNS THE PROGRAM WILL CONTINUE IN SEQUENCE. **CAUTION** DO NOT TURN POWER OFF/ON UNTIL THE MESSAGE 'POWER FAILED' HAS BEEN TYPED. THIS IS BECAUSE THE STACK MAY OVERFLOW.

8.6 EXECUTION TIME
EXECUTION TIME IS DEPENDENT ON THE AMOUNT OF MEMORY.

9.0 PROGRAM DESCRIPTION
THE PROGRAM VERIFIES EACH ADDRESS BY WRITING THE VALUE OF EACH ADDRESS INTO ITSELF STARTING AT LOCATION 20000 AND ENDING AT THE LAST LOCATION IN MEMORY. THE VALUE OF THE LAST LOCATION +2 IS TYPED ON THE TTY. NEXT THE VALUES WRITTEN ARE VERIFIED. TO COMPLETE THE ADDRESS TEST THE COMPLEMENT VALUE OF EACH MEMORY ADDRESS IS WRITTEN STARTING AT THE LAST MEMORY ADDRESS AND ENDING AT ADDRESS 20000. THE WRITTEN COMPLEMENT VALUES ARE THEN VERIFIED. THE NEXT PHASE OF TESTING INCLUDES READING, WRITING AND CHECKING MEMORY USING WORST CASE NOISE TEST PATTERN. A SUBTEST IS DEDICATED TO CHECKING THE PATTERN. THE TEST PROCEEDS BY EXERCISING EACH BANK OF MEMORY USING THE WORST CASE PATTERN. THE PROGRAM THEN CHECKS MEMORY USING RANDOM DATA (RANTST). THIS ROUTINE MOVES THE PROGRAM CODE THROUGHOUT MEMORY STARTING AT LOCATION 20000, AND RELOCATES THE DATA BY A 32(10) WORD OFFSET ON EACH SUBSEQUENT RELOCATION. I.E., FIRST RELOCATION IS TO 20000, NEXT IS TO 20100, THEN 20200, ETC. AFTER RELOCATION THE CODE MOVED IS CHECKED AGAINST THE ORIGINAL CODE (0-17776). WHEN THE RANDOM DATA TEST IS COMPLETE THE PROGRAM THEN SUCCESSIVELY ROTATES A 0 BIT (ROT0) AND A '1' BIT (ROT1) THROUGH ALL OF MEMORY. WHEN ALL TESTING IS COMPLETE THE PROGRAM INCREMENTS THE PASS COUNT (LOCATION 1000) AND RESTARTS BEGINNING WITH THE WORST CASE NOISE TESTS. AN ASTERISK (*) WILL BE TYPED ON COMPLETION OF EACH PASS, AND WHEN 6 PASSES HAVE BEEN COMPLETED THE PROGRAM WILL TYPE 'DEMJA DONE' AND RESTART THE PROGRAM BEGINNING WITH THE MEMORY ADDRESS TESTS.

```
%
.NLIST MD,MC
.LIST ME
.ABS
.MCALL $TYPE
.TITLE MAINDEC-11-DEMJA-B PDP11/70 MEMORY TEST
.SBTTL STARTING INST & DEFINITIONS
```

;COPYRIGHT 1973 1976 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

; THIS TEST CHECKS THAT ALL MEMORY ADDRESSES ARE UNIQUE USING ADDRESS TESTS
; AND CHECKS DATA RELIABILITY OF MEMORY USING WORST CASE NOISE TEST PATTERNS
; A RANDOM # PATTERN (PROGRAM CODE RELOCATED), A ROTATING 0 AND ROTATING
; 1 PATTERN.

```
;LOADING AND STARING INSTRUCTIONS
;LOAD ADDRESS 200 AND START
;NOTE: PROGRAM WILL RUN WORST CASE TEST PATTERNS IN LOWEST 4K
;THUS THE PROGRAM CANNOT BE RESTARTED AT 200 IF F _LOCATED. TO PREVENT
```



```

325      ;RELOCATION FROM OCCURRING DEPOSIT 200 INTO LOCATION 42 (NOT NECESSARY
326      ;IF LOADED VIA ACT11). THIS ACTION WILL PREVENT RELOCATION AND ALSO
327      ;INHIBIT TESTING MEMORY IN LOWEST 4K.
328      ;THIS PROGRAM ALSO RELOCATES THE ABS AND BOOT LOADERS TO ALLOW TESTING
329      ;OF MEMORY, TO RESTORE THE LOADERS RESTART AT 162.
330      ;STACK POINTER IS SET AT 500
331      ;AN ASTERISK '*' WILL BE PRINTED ON COMPLETION OF EACH PASS, AND
332      ;THE PROGRAM NAME WILL BE PRINTED WHEN TEST IS COMPLETE.
333
334      ;GENERAL REGISTER ASSIGNMENTS
335      000000      R0=%0
336      000001      R1=%1
337      000002      R2=%2
338      000003      R3=%3
339      000004      R4=%4
340      000005      R5=%5
341      000006      SP=%6
342      000007      PC=%7
343      000000      R10=%0
344      000001      R11=%1
345      000002      R12=%2
346      000003      R13=%3
347      000004      R14=%4
348      000005      R15=%5
349
350      ;STATUS REGISTER (PSW) BIT ASSIGNMENTS
351      000001      C=1      ;C BIT
352      000002      V=2      ;V BIT
353      000004      Z=4      ;Z BIT
354      000010      N=10     ;N BIT
355      000020      T=20     ;'T' BIT
356      000340      PRTY7=340 ;PRIORITY LEVEL 7
357      000200      PRTY4=200 ;PRIORITY LEVEL 4
358      000000      KM=000000 ;KERNEL MODE
359      040000      SM=040000 ;SUPERVISORY MODE
360      140000      UM=140000 ;USER MODE
361      000000      PKM=000000 ;PREVIOUS KERNEL MODE
362      010000      PSM=010000 ;PREVIOUS SUPERVISORY MODE
363      030000      PUM=030000 ;PREVIOUS USER MODE
364      004000      REG=004000 ;SELECT R10-R15
365
366      ;VECTOR ADDRESSES
367      000004      ERRVEC=4      ;ADDRESS OF ERROR VECTOR
368      000010      RESVEC=10     ;ADDRESS OF RESERVED INST. TRAP VECTOR
369      000014      TBITVEC=14    ;ADDRESS OF 'T' BIT TRAP VECTOR
370      000014      TRTVEC=14    ;ADDRESS OF 'TRACE' TRAP VECTOR
371      000014      BPTVEC=14    ;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
372      000020      IOTVEC=20     ;ADDRESS OF IOT TRAP VECTOR
373      000024      PFVEC=24     ;ADDRESS OF POWER FAIL TRAP VECTOR
374      000030      EMTVEC=30    ;ADDRESS OF EMT VECTOR
375      000034      TRAPVEC=34   ;ADDRESS OF TRAP VECTOR
376      000060      TKVEC=60     ;ADDRESS OF TTY KEYBOARD INTERRUPT VECTOR
377      000064      TP'EC=64     ;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
378      000240      PIRVEC=240   ;ADDRESS OF PIRQ VECTOR
379      000244      FPEVEC=244   ;ADDRESS OF FLOATING POINT INT. VECTOR
380      000250      MMVEC=250    ;ADDRESS OF MEM MGMT ERROR TRAP VECTOR

```


437 000000 UP=0 ;UP BIT IN PDR REGISTERS
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492

000000
000000
000002
000004
000006
000034
000036
000046
000052
000100
000104
000106
000110
000112
000162
000166
000172
000174
000200
000204
000250
000252
000254
000260
000262
000264

000000
000000
000000
001126
000002
000034
001204
000340
000046
004236
000052
040000
000100
004567
000746
000207
000000
000000
000162
012706
004767
000000
000401
000200
012706
000137
000250
000000
000000
012667
010546
010446
010346

000664
000500
002016
000500
002376
000016

. = 0
.WORD 0
.WORD 0
.WORD ERRTRP
.WORD RTI
. = TRAPVEC
.WORD ERROR
.WORD PRTY7
. = 46
LOGICAL
. = 52
40000
. = 100
CRLF: JSR RS, SPRINT
\$CRLF
RTS PC
RELF: .WORD 0
SAVPC2: .WORD 0
. = 162
PONE: MOV #500, SP ; STARTING ADDRESS TO RELOCATE LOADERS.
JSR PC, \$RLDR
HALT
BR PTWO
. = 200
PTWO: MOV #500, SP ; STARTING ADDRESS OF MEMORY TEST.
JMP @#START ; GO TO START OF TEST
. = 250
.WORD 0 ; MEMORY MANAGEMENT TRAP VECTOR.
.WORD 0
;
; ROUTINE TO SAVE REGISTERS ON THE STACK
; CALLED BY SAVE MACRO OR JSR PC, \$SAVR
\$SAVR: MOV (SP)+, 1\$; SAVE RETURN PC
MOV R5, -(SP)
MOV R4, -(SP)
MOV R3, -(SP)

; SPECIAL TRAP/INTERRUPT CATCHER IF PRO-
; GRAM HALTS AT 0 THEN ADDRESS WAS NOT
; LOADED PROPERLY FROM VECTOR.


```

493 000266 010246          MOV      R2,-(SP)
494 000270 010146          MOV      R1,-(SP)
495 000272 010046          MOV      R0,-(SP)
496 000274 012707          MOV      (PC)+,PC          ;RETURN
497 000276 000000          1$:      0                ;CONTAINS RETURN ADDRESS
498
499          ;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
500          ;CALLED BY RESTORE MACRO OR JSR PC,$RESTR
501 000300 012667 000016          $RESTR: MOV      (SP)+,1$          ;SAVE RETURN PC
502 000304 012600          MOV      (SP)+,R0
503 000306 012601          MOV      (SP)+,R1
504 000310 012602          MOV      (SP)+,R2
505 000312 012603          MOV      (SP)+,R3
506 000314 012604          MOV      (SP)+,R4
507 000316 012605          MOV      (SP)+,R5
508 000320 012707          MOV      (PC)+,PC          ;RETURN
509 000322 000000          1$:      0                ;CONTAINS RETURN ADDRESS
510
511          .SBTTL POWER FAIL ROUTINE
512          =502
513          ;POWER FAIL ROUTINE
514          ;THE POWER DOWN ROUTINE SAVES THE KEYBOARD STATUS, THE GENERAL REGISTERS
515          ;(R0-R5) AND MEM MGMT REGISTERS (KIPDR0-KIPDR7, KIPAR0-KIPAR7, SR3, SR2, SR0)
516          ;ON THE STACK AND SAVES THE STACK POINTER IN PFSTK BELOW.
517 000502 013746 177560          PDWN:  MOV      @#TKS,-(SP)          ;SAVE KEYBOARD STATUS
518 000506 004767 177542          JSR      PC,$SAVR          ;GO SAVE REGISTERS ON THE STACK
519 000512 005737 000762          TST      @#MMAVA          ;CHECK IF MEM MGMT IS AVAILABLE
520 000516 001421          BEQ      3$                ;BRANCH IF NOT AVAILABLE
521 000520 013746 177572          MOV      @#SR0,-(SP)          ;SAVE SR0
522 000524 013746 177576          MOV      @#SR2,-(SP)          ;SAVE SR2
523 000530 013746 172516          MOV      @#SR3,-(SP)          ;SAVE SR3
524 000534 012700 172300          MOV      #KIPDR0,R0          ;GET ADDRESS OF KIPDR0
525 000540 012702 000010          MOV      #8,R2
526 000544 010203          MOV      R2,R3
527 000546 012046          1$:      MOV      (R0)+,-(SP)          ;SAVE KIPDR0-KIPDR7
528 000550 077202          SOB      R2,1$
529 000552 012700 172340          MOV      #KIPAR0,R0          ;GET ADDRESS OF KIPAR0
530 000556 012046          2$:      MOV      (R0)+,-(SP)          ;SAVE KIPAR0-KIPAR7
531 000560 077302          SOB      R3,2$
532 000562 010627          3$:      MOV      SP,(PC)+          ;SAVE STACK PTR IN FOLLOWING LOCATION
533 000564 000000          PFSTK: .WORD      0          ;CONTAINS STACK PTR AFTER POWER FAIL
534 000566 012737 000576 000024          MOV      #PUP,@#PFVEC          ;SET POWER FAIL VECTOR TO PUP ROUTINE
535 000574 000000          HALT
536
537          ;POWER UP ROUTINE.
538 000576 000240          PUP:  NOP
539 000600 013706 000564          MOV      @#PFSTK,SP          ;SET STACK PTR
540 000604 005767 000152          TST      MMAVA          ;CHECK IF MEM MGMT IS AVAILABLE
541 000610 001421          BEQ      4$
542 000612 012700 172360          MOV      #KIPAR7+2,R0          ;GET ADDRESS OF KIPAR7+2
543 000616 012702 000010          MOV      #8,R2
544 000622 010203          MOV      R2,R3
545 000624 012640          1$:      MOV      (SP)+,-(R0)          ;RESTORE KIPAR7-KIPAR0
546 000626 077302          SOB      R3,1$
547 000630 012700 172320          MOV      #KIPDR7+2,R0          ;GET ADDRESS OF KIPDR7+2
548 000634 012640          2$:      MOV      (SP)+,-(R0)          ;RESTORE KIPDR7-KIPDR0

```



```

549 000636 077202          SUB      R2,2$
550 000640 012637 172516      MOV      (SP)+,3#SR3      ;RESTORE SR3
551 000644 012637 177576      MOV      (SP)+,3#SR2      ;RESTORE SR2
552 000650 012637 177572      MOV      (SP)+,3#SR0      ;RESTORE SR0
553 000654 005767 004522      4$:     TST      PARAVA      ;CHECK IF PARITY REGISTERS ARE ENABLED
554 000660 001402          BEQ      5$                ;BRANCH IF NOT
555 000662 004767 004444      JSR      PC,.MAMF         ;GO ENABLE PARITY REGISTERS
556 000666          5$:
557 000666 004767 177406      JSR      PC,$RESTR        ;RESTORE REGISTERS FROM STACK
558 000672 012637 177560      MOV      (SP)+,3#TKS
559 000676 012737 000502 000024      MOV      #PDWN,3#PFVEC    ;SET POWER FAIL TRAP TO PDWN ROUTINE
560 000704 005027          CLR      (PC)+
561 000706 000000      10$:     .WORD      0
562 000710 005267 177772      11$:     INC      10$            ;DELAY WAITING FOR TTY MOTOR
563 000714 100375          BPL      11$
564 000716 004567 000046      JSR      R5,$PRINT        ;GO TO PRINT ROUTINE
565 000722 000730          PWRFAIL
566 000724 000240      6$:     NOP
567 000726 000002          RTI                ;RETURN
568
569 000730 005015 047520 042527 PWRFAIL:.ASCII <15><12>'POWER FAILED'
570 000736 020122 040506 046111
571 000744 042105
572 000746 005015      000      $CRLF: .ASCIZ <15><12>
573
574
575          .SBTTL  TAGS & PRINT ROUTINE
576          .EVEN
577 000752 000000      ICNT:   .WORD      0                ;CONTAINS PASS COUNT
578 000754 000000      ICOUNT: .WORD      0                ;CONTAINS ITERATION PATTERN
579 000756 000000      ERCNT:  0                ;CONTAINS ERROR COUNT
580 000760 000000      LDDISP: 0                ;CONTAINS DISPLAY REGISTER IMAGE
581 000762 000000      MMAVA:  0                ;MEM MGMT AVAILABLE INDICATOR
582          ;0=NOT AVAIL,-1=AVAIL(18 BIT MODE)
583          ;-2=AVAIL(22 BIT MODE)
584
585 000764 000000      RELOCF: .WORD      0                ;CONTAINS RELOCATION FACTOR
586 000766 000000      COUNT:  .WORD      0                ;TEMPORARY WORKING LOCATION
587
588          ;ROUTINE TO PASS MESSAGE ADDRESS TO TYPE ROUTINE BELOW
589          ;CALL: JSR      R5,$PRINT
590          ;      MESSAGE ADDRESS
591 000770 000240          $PRINT: NOP
592 000772 012567 000016      MOV      (R5)+,1$        ;GET MESSAGE ADDRESS
593 000776 066767 177762 000010      ADD      RELOCF,1$        ;ADD RELOCATION FACTOR
594 001004 013746 177776          MOV      3#PSW,-(SP)     ;PUSH PSW ON THE STACK
595 001010 004767 000014      JSR      PC,.TYPE        ;CALL TYPE ROUTINE
596 001014 000000      1$:     .WORD      0                ;CONTAINS MESSAGE ADDRESS
597 001016 000205          RTS      R5                ;RETURN
598
599          ;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
600          ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
601          ;CALL: TYPE
602          ;      MESADR          ;MESADR IS FIRST ADDRESS OF ASCIZ STRING
603
604          ;TAGS USED BY THE TYPE ROUTINE BELOW

```



```

605 001020 000          SNULL: .BYTE 0          ;CONTAINS NULL CHARACTER
606 001021 002          SFILL:  .BYTE 2          ;CONTAINS # OF FILLER CHARACTERS
607 001022 000          STPFLG: .BYTE 0          ;CONTAINS TELEPRINTER AVAILABLE FLAG
608                                     ;0/377 = AVAIL/NOT AVAIL
609 001023 000          STKFLG: .BYTE 0          ;CONTAINS KEYBOARD AVAILABLE FLAG
610 001024 177564       STPS:  .WORD 177564       ;ADDRESS OF TELEPRINTER STATUS REGISTER
611 001026 177566       STPB:  .WORD 177566       ;ADDRESS OF TELEPRINTER DATA BUFFER
612 001030 010046       .TYPE: MOV R0,-(SP)      ;SAVE R0
613 001032 017600 000002 MOV R0,2(SP),R0      ;GET MESSAGE ADDRESS
614 001036 062766 000002 ADD #2,2(SP)          ;ADJUST RETURN PC
615
616 001044 112046       1$:  MOVB (R0)+,-(SP)    ;PUSH CHARACTER TO BE TYPED ONTO STACK
617 001046 001003       BNE 2$              ;BRANCH IF NOT THE TERMINATOR
618 001050 005726       TST (SP)+           ;POP TERMINATOR CHAR OFF THE STACK
619 001052 012600       MOV (SP)+,R0        ;RESTORE R0
620 001054 000002       RTI                          ;RETURN TO CALLER
621
622 001056 004767 000026 2$:  JSR PC,TYPIT      ;TYPE CHARACTER
623 001062 122726 000012 3$:  CMPB #12,(SP)+    ;CHECK IF CHARACTER WAS A LINE FEED
624 001066 001366       BNE 1$              ;BRANCH IF NOT LINE FEED
625 001070 016746 177724 MOV $NULL,-(SP)    ;GET # OF FILLERS REQUIRED AND FILLER
626                                     ;CHARACTER.
627
628 001074 105366 000001 4$:  DECB 1(SP)          ;DECREMENT FILLERS REQ. COUNT
629 001100 002770       BLT 3$              ;BRANCH IF NO MORE FILLERS ARE REQUIRED
630 001102 004767 000002 JSR PC,TYPIT      ;TYPE FILLER CHARACTER
631 001106 000772       BR 4$
632
633 001110 105777 177710 TYPIT: TSTB 2STPS    ;WAIT FOR OUTPUT DEVICE
634 001114 100375       BPL -4              ;
635 001116 116677 000002 177702 MOVB 2(SP),2STPB   ;OUTPUT CHARACTER
636 001124 000207       RTS PC
637
638 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
639 ;ERROR TRAP SERVICE ROUTINE
640 001126 005737 177570 ERRTRP: TST 2$SWR    ;CHECK IF HALT ON ERROR
641 001132 100001       BPL .+4           ;BRANCH IF NO HALT ON ERROR
642 001134 000000       HALT                          ;HALT
643 001136 005727       TST (PC)+           ;CHECK IF PREV TRAP TO 4 REPORTED
644 001140 000000       1$:  .WORD 0          ;CONTAINS ERROR REPORTED FLAG
645 001142 001013       BNE 2$              ;BRANCH IF NOT REPORTED
646 001144 010667 177770 MOV SP,1$         ;SET 'NOT REPORTED'
647 001150 011602       MOV (SP),R2        ;GET PC OFF STACK
648 001152 004767 000376 JSR PC,$FORMO    ;GO TO FORMAT ROUTINE
649 001156 004567 177606 JSR R5,$SPRINT   ;GO TO PRINT ROUTINE
650 001162 001460       TRAP4
651 001164 004567 177600 JSR R5,$SPRINT   ;GO TO PRINT ROUTINE
652 001170 002351       DIGITS
653 001172 000000       2$:  HALT                          ;ERROR! SECOND TRAP TO 4 OCCURRED
654                                     ;BEFORE FIRST WAS PRINTED
655 001174 005067 177740 CLR 1$
656 001200 000137 000200 JMP 2#200        ;RESTART AT 200
657
658 ;SBTTL ERROR SERVICE ROUTINE
659 ;ERROR SERVICE CALLED BY JSR PC,ERROR INSTRUCTION
660 ;OR HLT (A TRAP INST)
    
```


661	001204	000240			ERROR:	NOP		
662	001206	022767	017777	177542		CMP	#17777,ERCNT	;CHECK FOR MAX ERROR CNT
663	001214	001403				BEQ	4\$	
664	001216	062767	000001	177532		ADD	#1,ERCNT	;INCREMENT ERROR COUNT
665	001224	032737	001000	177570	4\$:	BIT	#BIT9,2\$SWR	;SWITCH 9 UP?
666	001232	001411				BEQ	5\$	
667	001234	042767	017777	177516		BIC	#17777,LDDISP	;SAVE RELOCATION BITS
668	001242	056767	177510	177510		BIS	ERCNT,LDDISP	;LOAD ERROR COUNT
669	001250	016737	177504	177570		MOV	LDDISP,2\$DISPLAY	;LOAD DISPLAY REGISTER
670	001256	005737	177570		5\$:	TST	2\$SWR	;HALT ON ERROR
671	001262	100002				BPL	.+6	
672	001264	000000				HALT		
673	001266	000470				BR	3\$	
674	001270	032737	020000	177570		BIT	#20000,2\$SWR	;PRINT OUT DESIRED?
675	001276	001051				BNE	1\$;BRANCH IF NO PRINTOUT
676	001300	004767	176750			JSR	PC,\$SAVR	;GO SAVE REGISTERS ON THE STACK
677	001304	016602	000014			MOV	14(SP),R2	;GET PC OF ERROR CALL
678	001310	004767	000240			JSR	PC,\$FORMO	;GO TO FORMAT ROUTINE
679	001314	004567	177450			JSR	R5,\$SPRINT	;GO TO PRINT ROUTINE
680	001320	001475				ERRPC		
681	001322	004567	177442			JSR	R5,\$SPRINT	;GO TO PRINT ROUTINE
682	001326	002351				DIGITS		
683	001330	016602	000004			MOV	4(SP),R2	;GET FAILING ADDRESS (IN R2)
684	001334	004767	000214			JSR	PC,\$FORMO	;GO TO FORMAT ROUTINE
685	001340	004567	177424			JSR	R5,\$SPRINT	;GO TO PRINT ROUTINE
686	001344	002327				ADDRESS		
687	001346	105767	003153			TSTB	PENFLG	;BRANCH IF PARITY ERROR DETECTED
688	001352	001017				BNE	11\$;BUT NOT FOUND
689	001354	105767	003144			TSTB	PEFLG	;BRANCH IF PARITY ERROR DETECTED
690	001360	001006				BNE	10\$;BUT FOUND
691	001362	004567	177402			JSR	R5,\$SPRINT	;GO TO PRINT ROUTINE
692	001366	001501				XMTDAT		
693	001370	010046				MOV	R0,-(SP)	;PUSH VALUE TO TYPED ONTO STACK
694	001372	004767	000416			JSR	PC,02A	;GO PRINT VALUE
695	001376				10\$:			
696	001376	004567	177366			JSR	R5,\$SPRINT	;GO TO PRINT ROUTINE
697	001402	001514				RECDAT		
698	001404	010346				MOV	R3,-(SP)	;PUSH VALUE TO BE TYPED ONTO STACK
699	001406	004767	000402			JSR	PC,02A	
700	001412	004767	176462		11\$:	JSR	PC,CRLF	
701	001416	004767	176656			JSR	PC,\$RESTR	;RESTORE REGISTERS FROM STACK
702	001422	032737	002000	177570	1\$:	BIT	#2000,2\$SWR	;RING BELL ON ERROR
703	001430	001403				BEQ	2\$	
704	001432	004567	177332			JSR	R5,\$SPRINT	;GO TO PRINT ROUTINE
705	001436	001527				BELL		
706	001440	005737	177570		2\$:	TST	2\$SWR	;HALT AFTER PRINT OUT
707	001444	100001				BPL	.+4	
708	001446	000000				HALT		
709	001450	010042			3\$:	MOV	R0,-(R2)	;RESTORE CORRECT DATA TO ADDRESS
710	001452	062702	000002			ADD	#2,R2	
711	001456	000002				RTI		
712								
713	001460	051124	050101	042520	TRAP4:	.ASCII	'TRAPPED TO 4'	
714	001466	020104	047524	032040				
715	001474	040						
716	001475	120	036503	000	ERRPC:	.ASCIZ	'PC='	


```

717 001501 107 047517 020104 XMTDAT: .ASCIZ 'GOOD DATA='
718 001506 040504 040524 000075
719 001514 041040 042101 042040 RECDAT: .ASCIZ 'BAD DATA='
720 001522 052101 036501 000
721 001527 007 000 BELL: .ASCIZ '<?>'
722 001531 120 051101 052111 PARREG: .ASCIZ '/PARITY ERROR REG=/'
723 001536 020131 051105 047522
724 001544 020122 042522 036507
725 001552 000
726 001554
727 .EVEN
:ROUTINE TO PLACE ASCII VALUE OF AN ADDRESS IN TO ADDRESS MESSAGE
728 001554 066767 177204 000014 $FORMO: ADD RELOC,11$+2
729 001562 066767 177176 000152 ADD RELOC,41$+2
730 001570 004767 176460 JSR PC,$SAVR ;GO SAVE REGISTERS ON THE STACK
731 001574 012704 002351 11$: MOV #DIGITS,R4 ;ADDRESS WHERE ASCII VALUES ARE STORED
732 001600 005003 CLR R3 ;WORKING & INDEX REGISTER
733 001602 162702 000002 SUB #2,R2 ;ADJUST ADDRESS
734 001606 010205 MOV R2,R5 ;SAVE
735 001610 010501 MOV R5,R1
736 001612 005767 177144 TST MMAVA ;CHECK IF MEM MGMT IS AVAILABLE
737 001616 001426 BEQ 1$ ;BRANCH IF NOT AVAILABLE
738 001620 032737 000001 177572 BIT #1,$#SRO ;IS MEM MGMT ENABLED
739 001626 001422 BEQ 1$
740 001630 042701 017777 BIC #17777,R1 ;SAVE PAR SELECTOR BITS
741 001634 000301 SWAB R1 ;SWAP BYTES
742 001636 006001 ROR R1
743 001640 006001 ROR R1 ;FORM INDEX VALUE
744 001642 006001 ROR R1
745 001644 006001 ROR R1
746 001646 017102 001774 MOV $PARTAB(1),R2 ;GET CONTENTS OF PAR
747 001652 012700 000006 MOV #6,R0 ;SHIFT COUNT
748 001656 006302 ASL R2 ;SHIFT KIPARI 6 PLACES LEFT
749 001660 006103 ROL R3 ;MSB'S GO INTO R3
750 001662 077003 SOB R0,-4
751 001664 042705 160000 BIC #160000,R5 ;CLEAR PAR SELECTOR BITS
752 001670 060502 ADD R5,R2 ;FORM 22 BIT ADDRESS
753 001672 005503 ADC R3 ;IN R2 & R3
754 001674 005001 1$: CLR R1
755 001676 012700 000005 MOV #5,R0
756 001702 006003 12$: ROR R3
757 001704 006002 ROR R2
758 001706 006001 ROR R1
759 001710 005300 DEC R0
760 001712 001373 BNE 12$
761 001714 012700 000010 MOV #8.,R0 ;DIGIT COUNT
762 001720 000405 BR 3$ ;PRINT FIRST DIGIT
763 001722 006301 2$: ASL R1
764 001724 006102 ROL R2
765 001726 006103 ROL R3
766 001730 005305 DEC R5
767 001732 001373 BNE 2$
768 001734 012705 000003 3$: MOV #3,R5 ;DIGIT SHIFT COUNT
769 001740 116324 002312 41$: MOVB DIGTAB(3),(4)+ ;LOAD DIGIT INTO MESSAGE
770 001744 005003 CLR R3 ;CLEAR INDEX
771 001746 005300 DEC R0 ;DEC DIGIT COUNT
772 001750 001364 BNE 2$
    
```



```

773 001752 004767 176322      JSR      PC,$RESTR      ;RESTORE REGISTERS FROM STACK
774 001756 046767 177002 177612  BIC      RELOCF,11$+2
775 001764 046767 176774 177750  BIC      RELOCF,41$+2
776 001772 000207                RTS      PC              ;RETURN
777
778 001774 172340      PARTAB: KIPAR0
779 001776 172342      KIPAR1
780 002000 172344      KIPAR2
781 002002 172346      KIPAR3
782 002004 172350      KIPAR4
783 002006 172352      KIPAR5
784 002010 172354      KIPAR6
785 002012 172356      KIPAR7
786
787 ;ROUTINE TO TYPE OCTAL VALUE PUSHED ONTO STACK
788 ;CALL: MOV      VALUE,-(SP)      ;PUSH VALUE ONTO STACK
789 ;      JSR      PC,02A          ;CALL ROUTINE
790
791 002014 004767 176234      02A:   JSR      PC,$SAVR      ;GO SAVE REGISTERS ON THE STACK
792 002020 016600 000016      MOV      16(SP),R0      ;GET VALUE
793 002024 012703 000006      MOV      #6,R3          ;COUNTER
794 002030 005002                CLR      R2              ;WORKING REGISTER
795 002032 006100      ROL      R0
796 002034 006102      ROL      R2
797 002036 062702 000260      15:   ADD      #260,R2          ;FORM ASCII VALUE
798 002042 010267 000040      MOV      R2,2$          ;MOVE CHAR TO TYPE LOCATION
799 002046 004567 176716      JSR      R5,$SPRINT      ;GO TO PRINT ROUTINE
800 002052 002106      2$:   CLR      R2
801 002054 005002      ROL      R0
802 002056 006100      ROL      R2
803 002060 006102      ROL      R0
804 002062 006100      ROL      R2
805 002064 006102      ROL      R0
806 002066 006100      ROL      R2
807 002070 006102      ROL      R0
808 002072 005303      DEC      R3
809 002074 001360      BNE      1$
810 002076 004767 176176      JSR      PC,$RESTR      ;RESTORE REGISTERS FROM STACK
811 002102 012616      MOV      (SP)+,(SP)
812 002104 000207      RTS      PC
813 002106 000000      25:   .WORD    0              ;CONTAINS CHARACTER TO BE TYPED
814
815 002110 000000      LODFLO: .WORD    0
816 ;ROUTINE TO SAVE ABS LOADER
817 002112 005767 177772      $LDR:  TST      LODFLO
818 002116 001401      BEQ      3$
819 002120 000207      RTS      PC
820 002122 012700 017776      3$:   MOV      #17776,R0
821 002126 012737 002140 000004  MOV      #2$,2$ERRVEC    ;SET TIME OUT TRAP VECTOR
822 002134 005720      TST      (R0)+
823 002136 000776      BR       -2
824 002140 022626      2$:   CMP      (SP)+,(SP)+
825 002142 022700 020000      CMP      #20000,R0      ;4K MACHINE?
826 002146 001417      BEQ      4$              ;YES--GET OUT
827 002150 162700 005672      SUB      #1500,+1*2,R0  ;POINT R0 BACK TO LOADER
828 002154 010067 000102      MOV      R0,$LDR1      ;SAVE FOR RESTORE ROUTINE
    
```



```

829 002160 012702 002734          MOV      #1500.,R2      ;WORD COUNT
830 002164 012703 010176          MOV      #LODAR,R3    ;WHERE LOADER IS TO BE STORED
831 002170 012023          1$:     MOV      (R0)+,(R3)+ ;STORE LOADER
832 002172 005302          DEC      R2
833 002174 001375          BNE     1$
834 002176 014367 000042          MOV      -(R3),LSTLOC ;SAVE LAST WORD OF LOADERS
835 002202 005367 177702          DEC      LODFLO
836 002206 000207          4$:     RTS      PC      ;RETURN
837
838          ;ROUTINE TO RESTORE LOADER
839 002210 005767 177674          $RLDR:  TST      LODFLO
840 002214 001001          BNE     2$
841 002216 000207          RTS      PC
842 002220 016705 000036          2$:     MOV      $LDR1,R5 ;GET FIRST ADDRESS OF WHERE LOADER IS
843          ;TO BE RESTORED
844 002224 012704 010176          MOV      #LODAR,R4    ;ADDRESS WHERE LOADER IS STORED
845 002230 012702 002734          MOV      #1500.,R2    ;WORD COUNT
846 002234 012425          1$:     MOV      (R4)+,(R5)+ ;RESTORE
847 002236 005302          DEC      R2
848 002240 001375          BNE     1$
849 002242 012745          MOV      (PC)+,-(R5) ;RESTORE LAST LOCATION (SAVED BY SAVE
850 002244 000000          LSTLOC: .WORD    0    ;LOADERS ROUTINE ABOVE)
851 002246 004567 176516          JSR     R5,$PRINT    ;GO TO PRINT ROUTINE
852 002252 002264          $LDRM:  $LDRM
853 002254 005067 177630          CLR     LODFLO
854 002260 000207          RTS      PC      ;RETURN TO CALLER
855
856 002262 000000          $LDR1:  .WORD    0    ;FIRST ADDRESS WHERE LOADERS ARE TO BE
857          ;RESTORED TO
858 002264 047514 042101 051105          $LDRM:  .ASCIZ  'LOADER IS RESTORED'<15><12>
859 002272 044440 020123 042522
860 002300 052123 051117 042105
861 002306 005015 000
862 002312
863          ;DIGIT TABLE
864 002312 030460          DIGTAB: "01
865 002314 031462          "23
866 002316 032464          "45
867 002320 033466          "67
868
869          ;MESSAGES
870 002322 040514 052123 040          LST:    .ASCII  'LAST '
871 002327 0115 046505 051117          ADRESS: .ASCII  'MEMORY ADDRESS IS '
872 002334 020131 042101 051104
873 002342 051505 020123 051511
874 002350 040
875 002351 060 030060 030060          DIGITS: .ASCII  '00000000'
876 002356 030060 060
877 002361 040 000          SPACE1: .ASCIZ  ' '
878 002363 120 051501 036523          PASSMG: .ASCII  'PASS='
879 002370 020040 000          PASSNM: .ASCIZ  ' '
880 002374
881 002374 000000          PLACE:  .WORD    0
882          .SBTTL  MEMORY ADDRESS TESTS
883
884          ;THIS TEST ADDRESS MEMORY UP TO 128K AND PROVES 'UNIQUENESS' OF ALL
    
```



```

885 ;MEMORY ADDRESS IN A 32K SEGMENT. THE TEST WRITES INTO EACH MEMORY
886 ;ADDRESS THE VALUE OF THAT ADDRESS AND THEN CHECKS FOR THE CORRECT
887 ;DATA IN EACH ADDRESS.
888 ;THE TWELVE MOST SIGNIFICANT BITS OF THE LAST AVAILABLE MEMORY ADDRESS
889 ;IS STORED IN R5.
890 ;STARTING INSTRUCTIONS
891 ;LOAD ADDRESS=200
892 ;PRESS START
893 ;STACK POINTER IS AT 500
894 ;*****RESTART AT 162 TO RESTORE LOADER*****
895 ;MEMORY ADDRESS TEST
896 002376 012737 002440 000212 START: MOV #START1, R212 ;CHANGE START ADDRESS
897 002404 012706 000500 MOV #STKPTR, SP ;SET UP STACK PTR
898 002410 004767 177476 JSR PC, $LDR ;GO SAVE MONITOR & LOADERS
899 002414 004567 176350 JSR R5, $PRINT ;GO TO PRINT ROUTINE
900 002420 007516 RESLDR
901 002422 005037 000756 CLR R#ERCNT ;CLEAR ERROR COUNT
902 002426 005037 000760 CLR R#LDDISP ;CLEAR DISPLAY REGISTER STORAGE LOCN
903 002432 013737 000760 177570 MOV R#LDDISP, R#DISPLAY ;CLEAR DISPLAY REGISTER
904 002440 012706 000500 START1: MOV #STKPTR, SP ;SET STACK PTR
905 002444 005037 004524 CLR R#PEFLG ;CLEAR PARITY ERROR INDICATORS
906 002450 012727 002440 MOV #START1, (PC)+ ;LOAD PARITY ERROR RESTART ADDRESS
907 002454 000000 PERSTRT: WORD 0 ;CONTAINS RESTART ADDRESS AFTER PAR ERR
908 002456 005037 000752 CLR R#ICNT ;CLEAR PASS COUNT
909 002462 005037 000764 CLR R#RELOCF ;CLEAR RELOCATION FACTOR
910 002466 012737 000502 000024 MOV #PDWN, R#PFVEC ;SET POWER FAIL TRAP VECTOR
911 002474 005037 000026 CLR R#PFVEC+2
912
913 ;CHECK IF MEMORY MANAGEMENT IS AVAILABLE
914 002500 005067 176256 CLR MAVA ;CLEAR MEM MGMT AVAILABLE INDICATOR
915 002504 032737 010000 177570 BIT #BIT12, R#SWR ;CHECK IF TO RUN WITH MEM MGMT
916 002512 001020 BNE 1$ ;DO NOT USE MEM MGMT IF SW12 WAS SET
917 002514 012737 002554 000004 MOV #1$, R#ERRVEC ;SET TIME OUT TRAP
918 002522 005037 177572 CLR R#SRO ;REFERENCE MEM MGMT
919 002526 005167 176230 COM MAVA ;SET INDICATOR TO -1 IF AVAILABLE
920 002532 012737 000020 172516 MOV #20, R#SR3 ;SET 22 BIT MODE
921 002540 022737 000020 172516 CMP #20, R#SR3 ;DID IT SET?
922 002546 001002 BNE 1$ ;NO--BRANCH
923 002550 006367 176206 ASL MAVA ;YES--SET INDICATOR TO -2
924 002554 004767 002552 1$: JSR PC, .MAMF ;GO ENABLE PARITY ACTION
925
926
927 ;ROUTINE TO WRITE VALUE OF MEMORY ADDRESS INTO MEMORY ADDRESS
928 ;FOR EXAMPLE ROUTINE WRITES 20000 INTO LOCATION 20000
929 002560 012737 002620 000004 WRTUP: MOV #DONEO, R#ERRVEC ;SET TIME OUT TRAP VECTOR
930 002566 010701 MOV PC, R1 ;LOAD TRACE REGISTER
931 002570 004767 002716 JSR PC, LDMMO
932 002574 012737 005614 000250 MOV #MMABTO, R#MMVEC ;SET MEM MGMT ABORT VECTOR
933 002602 012702 020000 MOV #20000, R2 ;FIRST ADDRESS
934 002606 010203 MOV R2, R3 ;LOAD CONSTANT
935 002610 010322 1$: MOV R3, (R2)+ ;WRITE VALUE OF ADDRESS INTO ADDRESS
936 002612 062703 000002 ADD #2, R3 ;NEXT VALUE
937 002616 000774 BR 1$ ;WRITE UNTIL DONE
938
939 002620 012706 000500 DONEO: MOV #STKPTR, SP ;SET STACK PTR
940 002624 004767 176724 JSR PC, $FORMO ;GO TO FORMAT ROUTINE
    
```



```

941 002630 004567 176134      JSR    R5,$SPRINT      ;GO TO PRINT ROUTINE
942 002634 002322              LST
943 002636 004767 175236      JSR    PC,CRLF
944
945      ;ROUTINE TO CHECK THAT VALUE OF MEMORY ADDRESS WAS WRITTEN CORRECTLY
946 002642 010701              MOV    PC,R1           ;LOAD TRACE REGISTER
947 002644 012702 020000      MOV    #20000,R2      ;SET R2
948 002650 012737 002724 000004  MOV    #DONE1,@#ERRVEC ;SET TIME OUT TRAP
949 002656 010200              MOV    R2,R0
950 002660 162700 000002      SUB    #2,R0           ;SUBTRACT 2
951 002664 004767 002622      JSR    PC,LDMMO
952 002670 062700 000002      1$:   ADD    #2,R0
953 002674 012203              MOV    (R2)+,R3       ;GET WRITTEN VALUE
954 002676 020003              CMP    R0,R3          ;CHECK
955 002700 001402              BEQ    2$
956 002702 104400              HLT
957 002704 000771              BR     1$             ;ERROR! TO DETERMINE WHICH ADDRESS WAS
958 002706 005142              2$:   COM    -(R2)
959 002710 005112              COM    (R2)
960 002712 012203              MOV    (R2)+,R3
961 002714 020003              CMP    R0,R3
962 002716 001764              BEQ    1$
963 002720 104400              HLT
964      ;WRITTEN IMPROPERLY EXAMINE R2. NEXT EXAMINE MEM MGMT REGISTER KIPARI
965      ;(IF MEM MGMT IS AVAILABLE). ADD R2 AND KIPARI TOGETHER AS SHOWN BELOW
966
967      :      R2-2      0 00X XXX XXX XXX XXX
968      :      KIPARI(772342) Y YYY YYY YYY YYY YYY
969      :      ADDRESS      Z ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ ZZZ
970
971 002722 000762              BR     1$
972 002724 012706 000500      DONE1: MOV    #STKPTR,SP ;SET STACK PTR
973 002730 010701              MOV    PC,R1          ;LOAD TRACE REGISTER
974
975      ;ROUTINE TO WRITE 1'S COMPLEMENT VALUE OF ADDRESS INTO ADDRESS
976      ;FOR EXAMPLE ROUTINE WRITES 157777 INTO ADDRESS 20000
977
978 002732 005767 176024      TST    MAVA           ;MEMORY MAGNAGEMENT AVAILABLE?
979 002736 001420              BEQ    3$
980 002740 013703 172342      MOV    @#KIPARI,R3   ;FIND LAST ADDRESS IF MEM MANAGE USED
981 002744 006303              ASL    R3
982 002746 006303              ASL    R3
983 002750 006303              ASL    R3
984 002752 006303              ASL    R3
985 002754 006303              ASL    R3
986 002756 006303              ASL    R3
987 002760 010246              MOV    R2,-(SP)      ;DEVELOP COMPLEMENT OF LAST ADDRESS
988 002762 042716 020000      BIC    #20000,(SP)   ;SAVE BITS IF MEMORY IS NOT A MULTIPLE OF 4K
989 002766 062603              ADD    (SP)+,R3
990 002770 012737 005646 000250  MOV    #MMABT1,@#MMVEC ;SET ABORT VECTOR
991 002776 000403              BR     2$
992 003000 162702 000072      3$:   SUB    #2,R2         ;R2=LAST ADDRESS
993 003004 010203              MOV    R2,R3
994 003006 005103              2$:   COM    R3          ;COMPLEMENT VALUE IN R3
995 003010 062703 000002      1$:   ADD    #2,R3
996 003014 010342              MOV    R3,-(R2)     ;WRITE COMPLIMENT VALUE INTO ADDRESS
    
```



```

997 003016 102403          BVS  DONE3
998 003020 020227 017776    CMP  R2,#17776
999 003024 001371          BNE  1$
1000
1001          ;SET UP TO CHECK COMPLEMENT DATA WRITTEN DOWN
1002 003026 000240          DONE3: NOP
1003 003030 010701          MOV  PC,R1          ;LOAD TRACE REGISTER
1004 003032 005767 175724    TST  MMVA          ;CHECK IF MM IS AVAIL
1005 003036 001406          BEQ  1$
1006 003040 012737 000200 172342    MOV  #200,#KIPARI  ;INIT KIPARI
1007 003046 012737 005614 000250    MOV  #MMABTO,#MMVEC ;SET ABORT VECTOR
1008 003054 012737 003114 000004 1$:  MOV  #DONE4,#ERRVEC
1009 003062 012702 020000          MOV  #20000,R2     ;FIRST ADDRESS
1010 003066 010200          MOV  R2,R0
1011 003070 005100          COM  R0            ;FIRST DATA (COM OF ADDRESS)
1012 003072 062700 000002          ADD  #2,R0
1013 003076 162700 000002 2$:  SUB  #2,R0
1014 003102 012203          MOV  (R2)+,R3     ;GET VALUE
1015 003104 020003          CMP  R0,R3        ;CHECK
1016 003106 001773          BEQ  2$
1017 003110 104400          HLT
1018 003112 000771          BR   2$
1019 003114 000240          DONE4: NOP
1020
1021          ;ROUTINE TO WRITE BANK # INTO ALL ADDRESSES IN A 4K BANK
1022 003116 012737 003164 000004    MOV  #DONE4A,#ERRVEC ;SET TIME OUT TRAP VECTOR
1023 003124 010701          MOV  PC,R1
1024 003126 004767 002360          JSR  PC,LDMMO
1025 003132 012737 005614 000250    MOV  #MMABTO,#MMVEC
1026 003140 012702 020000          MOV  #20000,R2
1027 003144 005000          CLR  R0
1028 003146 005200 1$:  INC  R0            ;R0 WILL BE DATA WRITTEN
1029 003150 012704 010000          MOV  #4096,R4     ;SET 4K COUNTER
1030 003154 010022 2$:  MOV  R0,(R2)+     ;WRITE BANK # INTO ALL ADDRESSES
1031 003156 005304          DEC  R4
1032 003160 001375          BNE  2$
1033 003162 000771          BR   1$
1034
1035 003164 022626          DONE4A: CMP  (SP)+,(SP)+ ;ADJUST STACK PTR
1036
1037          ;CHECK THAT DATA WRITTEN ABOVE CAN BE READ
1038 003166 012737 003234 000004    MOV  #DONE4B,#ERRVEC
1039 003174 010701          MOV  PC,R1
1040 003176 004767 002310          JSR  PC,LDMMO
1041 003202 012702 020000          MOV  #20000,R2
1042 003206 005000          CLR  R0
1043 003210 005200 1$:  INC  R0
1044 003212 012704 010000          MOV  #4096,R4
1045 003216 012203 2$:  MOV  (R2)+,R3
1046 003220 020003          CMP  R0,R3
1047 003222 001401          BEQ  .+4
1048 003224 104400          HLT
1049 003226 005304          DEC  R4
1050 003230 001372          BNE  2$
1051 003232 000766          BR   1$
1052 003234 022626          DONE4B: CMP  (SP)+,(SP)+

```



```

1053
1054 ;ROUTINE TO WRITE CONSTANT DATA INTO 4K
1055 ;BANK STARTING WITH LAST MEMORY LOCATION
1056 003236 010701          MOV      PC,R1
1057 003240 012737 005646 000250  MOV      #MMABT1, @#MMVEC
1058 003246 162702 000002          SUB      #2,R2
1059 003252 005000          CLR      R0
1060 003254 005300          1$: DEC      R0
1061 003256 012704 010000          MOV      #4096.,R4
1062 003262 010042          2$: MOV      R0,-(R2)
1063 003264 102406          BVS     DONE4C
1064 003266 020227 017776          CMP      R2,#17776 ;CHECK IF DONE
1065 003272 001403          BEQ     DONE4C
1066 003274 005304          DEC      R4
1067 003276 001371          BNE     2$
1068 003300 000765          BR      1$
1069
1070 003302 012737 003414 000004  DONE4C: MOV     #DONE4D, @#ERRVEC
1071 003310 010701          MOV     PC,R1
1072 003312 004767 002174          JSR    PC,LDMMO
1073 003316 012737 005614 000250  MOV     #MMABT0, @#MMVEC ;SET ABORT VECTOR
1074 003324 012702 020000          MOV     #20000,R2
1075 003330 022704 010000          1$:  CMP     #4096.,R4 ;CHECK IF WRITE ABOVE STARTED ON
1076                                     ;4K BOUNDARY
1077                                     BEQ     2$
1078 003334 001415          MOV     (R2)+,R3
1079 003336 012203          CMP     R0,R3
1080 003340 020003          CMP     R0,R3
1081 003342 001402          BEQ     4$
1082 003344 104400          HLT
1083 003346 000406          BR      5$
1084 003350 005142          4$:  COM     -(R2)
1085 003352 005112          COM     (R2)
1086 003354 012203          MOV     (R2)+,R3
1087 003356 020003          CMP     R0,R3
1088 003360 001401          BEQ     5$
1089 003362 104400          HLT
1090 003364 005204          5$:  INC     R4
1091 003366 001360          BNE     1$
1092 003370 005200          2$:  INC     R0
1093 003372 012704 010000          MOV     #4096.,R4
1094 003376 012203          3$:  MOV     (R2)+,R3
1095 003400 020003          CMP     R0,R3
1096 003402 001401          BEQ     .+4
1097 003404 104400          HLT
1098 003406 005304          DEC     R4
1099 003410 001372          BNE     3$
1100 003412 000766          BR      2$
1101
1101 003414 022626          DONE4D: CMP     (SP)+,(SP)+
1102 003416 005737 000042          TST     @#42 ;BRANCH IF PROGRAM WAS NOT
1103 003422 001406          BEQ     BEGIN1 ;LOADED VIA ACT11 IN QV OR AA MODES
1104 003424 005767 000610          TST     LOGICAL+2 ;BRANCH IF NOT IN QV MODE
1105 003430 100003          BPL     BEGIN1
1106 003432 012737 000001 004170  MOV     #1, @#ENDCT ;SET ENDCT TO DO 1 PASS ONLY IN QV
1107                                     .SBTTL  WORST CASE NOISE TESTS
1108                                     ;THIS TEST WRITES MEMORY WORST CASE NOISE TEST PATTERNS THROUGHOUT
    
```



```

1109 ;MEMORY AND CHECKS THAT THEY CAN BE WRITTEN AND READ.
1110 ;SET UP TRAP VECTORS
1111 003440 012706 000500 BEGIN1: MOV #STKPTR, SP ;SET STACK PTR
1112 003444 004767 001662 JSR PC, MAMF ;GO ENABLE PARITY ACTION
1113 003450 004767 003740 JSR PC, CKSWR ;GO CHECK SWITCHES
1114 003454 005027 CLR (PC)+ ;SET INDICATOR TO WRITE NORMAL 3X9 PAT
1115 003456 000000 PARPAT: .WORD 0
1116 003460 022767 177776 175274 CMP #-2, MMAVA ;22 BIT MODE
1117 003466 001002 BNE DONE6 ;NO--BRANCH
1118 003470 004767 001770 JSR PC, MARGIN ;YES--GO SETUP MARGINS
1119
1120
1121 ;WRITE 3 XOR 9 TEST PATTERN STARTING AT ADDRESS 20000
1122 ;NOTE PATTERN IS NORMAL 3 XOR 9 IF NO PARITY MEMORY IS AVAILABLE,
1123 ;AND IS A MODIFIED PATTERN IF PARITY MEMORY IS AVAILABLE.
1124 ;THE CONTENTS OF PARPAT IF 0/NOT 0 INDICATE IF NORMAL/MODIFIED PATTERN
1125 ;IS BEING USED IN TESTS BELOW.
1126 003474 012706 000500 DONE6: MOV #STKPTR, SP ;SET STACK PTR
1127 003500 010701 MOV PC, R1 ;UPDATE TRACE REGISTER
1128 003502 012737 003522 000004 MOV #DONE7, J#ERRVEC ;SET TIME OUT TRAP VECTOR
1129 003510 012746 000001 MOV #1, -(SP) ;PUSH STARTING BANK # ON STACK
1130 003514 005046 CLR -(SP) ;PUSH # OF 256. WORD BLOCKS TO WRITE
1131 003516 004767 002340 JSR PC, .3X9 ;CALL ROUTINE TO WRITE 3XOR9 PATTERN
1132
1133 ;CHECK 3 XOR 9 TEST PATTERN WRITTEN ABOVE
1134 003522 012737 001126 000004 DONE7: MOV #ERRTRP, J#ERRVEC
1135 003530 016600 000006 MOV 6(SP), R0 ;GET # OF 256. WORD BLOCKS WRITTEN
1136 003534 005400 NEG R0 ;FORM TWO'S COMPLEMENT
1137 003536 010027 MOV R0, (PC)+ ;SAVE # OF 256 WORD BLOCKS
1138 003540 000000 WDS.256: .WORD 0 ;CONTAINS # OF 256 WORD BLOCKS IN MEM.
1139 003542 012706 000500 MOV #STKPTR, SP ;SET STACK PTR
1140 003546 010701 MOV PC, R1 ;SET SCOPE PTR
1141 003550 012746 000001 MOV #1, -(SP) ;PUSH BANK # ON THE STACK
1142 003554 010046 MOV R0, -(SP) ;PUSH # OF 256. WORD BLOCKS TO WRITE
1143 003556 004767 002520 JSR PC, .3X9 ;GO CHECK DATA WRITTEN
1144
1145 ;SETUP TO RUN MODIFIED 3 XOR 9 PATTERN IF PARITY MEMORY IS AVAILABLE
1146 003562 022767 177776 175172 CMP #-2, MMAVA
1147 003570 001403 BEQ 1$
1148 003572 005737 005402 TST J#PARAVA ;BRANCH IF PARITY MEMORY IS NOT AVAIL
1149 003576 001406 BEQ DONE8
1150 003600 005737 003456 1$: TST J#PARPAT ;BRANCH IF PARITY PAT JUST WRITTEN
1151 003604 001003 BNE DONE8
1152 003606 010637 003456 MOV SP, J#PARPAT ;SET INDICATOR TO WRITE 3X9 PAR PAT
1153 003612 000730 BR DONE6 ;REPEAT TEST USING MODIFIED 3X9 PATTERN
1154
1155 ;WRITE 8 XOR 13 TEST PATTERN STARTING AT ADDRESS 40000
1156 003614 012706 000500 DONE8: MOV #STKPTR, SP ;SET STACK PTR
1157 003620 012737 003642 000004 MOV #DONE9, J#ERRVEC ;SET TIME OUT TRAP VECTOR
1158 003626 010701 MOV PC, R1 ;UPDATE TRACE REGISTER
1159 003630 012746 000002 MOV #2, -(SP) ;PUSH STARTING BANK # ON THE STACK
1160 003634 005046 CLR -(SP) ;PUSH # OF BANKS TO WRITE ON THE STACK
1161 003636 004767 003234 JSR PC, .8X13 ;GO TO ROUTINE TO WRITE DATA
1162
1163 ;CHECK 8 XOR 13 TEST PATTERN WRITTEN ABOVE
1164 003642 012706 000500 DONE9: MOV #STKPTR, SP ;SET STACK PTR

```



```

1165 003646 010701          MOV      PC,R1          ;UPDATE TRACE REGISTER
1166 003650 012737 001126 000004  MOV      #ERRTRP,#ERRVEC
1167 003656 012746 000002          MOV      #2,-(SP)
1168 003662 005404          NEG      R4
1169 003664 042704 000001          BIC      #1,R4          ;SET 4K BANK COUNT TO 8K INCREMENT
1170 003670 001403          BEQ      DONE10        ;DO NOT CHECK IF ONLY 12K
1171 003672 010446          MOV      R4,-(SP)
1172 003674 004767 003204          JSR      PC,..8X13     ;GO CHECK 8 XOR 13 PATTERN WRITTEN ABOVE
1173
1174
1175 003700 000005          DONE10: RESET          ;DISABLE MEM MGMT AND PARITY ACTION
1176
1177
1178          .SBTTL RANDOM DATA,ROTATING 1/0 TESTS
1179          :RANDOM DATA TEST. THIS TEST MOVES THE PROGRAM CODE THROUGHOUT MEMORY
1180 003702 010701          RANTST: MOV      PC,R1          ;SET TRACE POINTER
1181 003704 012737 004042 000004  MOV      #7$,#ERRVEC   ;SET TIME OUT TRAP
1182 003712 005767 175044          TST      MAVA          ;CHECK IF MEM MGMT IS AVAILABLE
1183 003716 001412          BEQ      1$           ;BRANCH IF NOT AVAILABLE
1184 003720 004767 001566          JSR      PC,LDMMO      ;GO SET UP MEM MGMT
1185 003724 105237 172301          INCB     #KIPDR0+1     ;ALLOW 4K ADDRESSING IN FIRST 4K
1186 003730 012737 077406 172304  MOV      #200*256.-400+UP#RW,#KIPDR2 ;SET KIPDR2=RW UP 200 BLOCKS
1187 003736 012737 000400 172344  MOV      #400,#KIPAR2
1188 003744 012702 020000          1$: MOV      #20000,R2   ;SET 'TO' ADDRESS POINTER
1189 003750 005004          CLR      R4           ;SET 'FROM' ADDRESS POINTER
1190 003752 012705 004000          2$: MOV      #2048.,R5   ;SET 4K WORD COUNT
1191 003756 012422          3$: MOV      (R4)+,(R2)+ ;MOVE CODE
1192 003760 012422          MOV      (R4)+,(R2)+
1193 003762 005305          DEC      R5           ;DECREMENT 4K WORD COUNTER
1194 003764 001374          BNE      3$
1195
1196 003766 012705 005405          4$: MOV      #4096.-PLACE+1,R5 ;SET 4K WORD COUNTER
1197 003772 014400          MOV      -(R4),R0     ;GET 'GOOD' DATA
1198 003774 014203          MOV      -(R2),R3     ;GET 'BAD' DATA
1199 003776 020003          CMP      R0,R3        ;COMPARE 'GOOD' & 'BAD' DATA
1200 004000 001403          BEQ      5$
1201 004002 005722          TST      (R2)+        ;STEP ADDRESS FOR ERROR ROUTINE
1202 004004 104400          HLT                      ;REPORT ERROR
1203 004006 005742          TST      -(R2)        ;RESTORE ADDRESS POINTER
1204 004010 005305          5$: DEC      R5        ;DECREMENT 4K WORD COUNTER
1205 004012 001367          BNE      4$          ;LOOP UNTIL 4K WORDS CHECKED
1206
1207 004014 005767 174742          TST      MAVA          ;CHECK IF MEM MGMT IS AVAILABLE
1208 004020 001405          BEQ      6$           ;BRANCH IF NOT AVAILABLE
1209 004022 005237 172342          INC      #KIPAR1
1210 004026 005237 172344          INC      #KIPAR2
1211 004032 000744          BR       1$
1212 004034 062702 000100          6$: ADD      #64.,R2    ;STEP ADDRESS
1213 004040 000744          BR       2$
1214 004042 012706 000500          7$: MOV      #STKPTR,SP ;RESET STACK PTR
1215 004046 012737 001126 000004  MOV      #ERRTRP,#ERRVEC ;RESTORE ERROR TRAP VECTOR
1216
1217          :ROTATING 0 TEST. THIS TEST ROTATES A SINGLE '0' THROUGH MEMORY
1218 004054 012767 177777 003276  ROTO: MOV      #-1.,CONST ;SET CONSTANT =177777
1219 004062 012746 000001          MOV      #1,-(SP)     ;SET BANK #1
1220 004066 016746 177446          MOV      WDS.256,-(SP) ;GET # OF 256. WORD BLOCKS IN MEMORY

```



```

1221 004072 004767 003244      JSR    PC,WRTPAT      ;GO WRITE 1'S THROUGHOUT MEMORY
1222 004076 010701              MOV    PC,R1          ;SET SCOPE PTR
1223 004100 012746 000001      MOV    #1,-(SP)       ;SET STARTING BANK #
1224 004104 016746 177430      MOV    WDS.256,-(SP)  ;SET # OF 256. WORD BLOCKS TO CHECK
1225 004110 004767 002776      JSR    PC,.ROTO      ;GO TO ROTATE 0 ROUTINE
1226
1227                          ;ROTATING 1 TEST THIS TEST ROTATES A SINGLE '1' BIT THROUGH ALL OF
1228                          ;MEMORY
1229 004114 005067 003240      ROT1:  CLR    .CONST   ;CLEAR CONSTANT
1230 004120 012746 000001      MOV    #1,-(SP)       ;PUSH STARTING BANK ONTO STACK
1231 004124 016746 177410      MOV    WDS.256,-(SP)  ;AND # OF 256. WORD BLOCKS IN MEMORY
1232 004130 004767 003206      JSR    PC,WRTPAT      ;GO WRITE 0'S THROUGHOUT MEMORY
1233 004134 010701              MOV    PC,R1          ;SET SCOPE PTR
1234 004136 012746 000001      MOV    #1,-(SP)       ;SET STARTING BANK #
1235 004142 016746 177372      MOV    WDS.256,-(SP)  ;SET # OF 256. WORD BLOCKS TO CHECK
1236 004146 004767 003034      JSR    PC,.ROT1      ;GO ROTATE A '1' BIT THROUGHOUT MEMORY
1237
1238                          ;END OF CYCLE
1239 004152 000005      END:   RESET
1240 004154 010701              MOV    PC,R1          ;UPDATE TRACE REGISTER
1241 004156 012706 000500      MOV    #STKPTR,SP    ;SET STACK PTR
1242 004162 005237 000752      INC    @#ICNT         ;INCREMENT PASS COUNT
1243 004166 022737              CMP    (PC)+,@(PC)+  ;CHECK FOR LAST PASS
1244 004170 000006      ENDCT: .WORD    6      ;MAKE 5 PASSES
1245 004172 000752              .WORD    ICNT        ;PASS COUNT ADDRESS
1246 004174 001405      BEQ    DONE          ;BRANCH IF LAST PASS COMPLETED
1247 004176 004567 174566      JSR    RS,$PRINT     ;GO TO PRINT ROUTINE
1248 004202 010104      ASTERISK
1249 004204 000137 003440      JMP    @#BEGIN1
1250 004210
1251 004210 004567 174554      DONE:  JSR    RS,$PRINT     ;GO TO PRINT ROUTINE
1252 004214 010106      ENDMSG
1253 004216 105737 177564      TSTB  @#TPS          ;WAIT FOR BELL TO RING
1254 004222 100375      BPL    -4
1255 004224 013700 000042      MOV    @#42,R0       ;GET DECTAPE MONITOR RETURN ADDRESS
1256 004230 001406      BEQ    FINISH
1257 004232 004767 175752      JSR    PC,$RLDR     ;RESTORE MONITOR & LOADERS
1258 004236 004710      LOGICAL: JSR    PC,(R0) ;GO TO DECTAPE MONITOR
1259 004240 000240      NOP
1260 004242 000240      NOP
1261 004244 000240      NOP
1262 004246 000167 176166      FINISH: JMP    START1
1263
1264                          .SBTTL PROGRAM SUBROUTINES
1265                          .SBTTL RELOCATION ROUTINES
1266                          ;ROUTINE TO RELOCATE PROGRAM CODE
1267 004252 012500      RELOC: MOV    (R5)+,R0    ;GET FROM ADDRESS
1268 004254 011502      MOV    (R5),R2       ;GET TO ADDRESS
1269 004256 010203      MOV    R2,R3
1270 004260 062703 017776      ADD    #17776,R3     ;MOVES 4K
1271 004264 012737 004334 000004      MOV    #4$,@#ERRVEC ;SET TIME OUT TRAP
1272 004272 005004      CLR    R4           ;CLEAR RELOCATION SUCCESSFUL INDICATOR
1273 004274 005723      TST   (R3)+         ;CHECK IF MEMORY IS AVAILABLE
1274 004276 012022      IS:   MOV    (R0)+,(R2)+ ;RELOCATE
1275 004300 020203      CMP    R2,R3        ;RELOCATION COMPLETE?
1276 004302 001375      BNE   IS

```



```

1277 004304 011503          MOV      (R5),R3
1278 004306 020203      2$:    CMP      R2,R3
1279 004310 001413          BEQ      5$          ;BRANCH IF DONE
1290 004312 024042          CMP      -(R0),-(R2) ;CHECK THAT DATA WAS RELOCATED PROPERLY
1281 004314 001774          BEQ      2$
1282 004316 005703          TST      R3          ;CHECK IF RELOCATING BACK TO 000000
1283 004320 001403          BEQ      3$
1284 004322 104400          HLT
1285
1286 004324 000000          HALT
1287 004326 000767          BR      2$          ;CONTINUE RELOCATING AT YOUR PERIL
1288 004330 000000      3$:    HALT          ;ERROR! CANNOT RELOCATE CODE BACK TO
1289
1290 004332 000777          BR
1291 004334 022626      4$:    CMP      (SP)+,(SP)+ ;RESTORE STACK PTR
1292 004336 005104          COM      R4
1293 004340 000240      5$:    NOP
1294 004342 012702 000764          MOV      #RELOCF,R2 ;GET ADDRESS OF RELOCATION FACTOR
1295 004346 061502          ADD      (R5),R2    ;ADD FACTOR
1296 004350 012512          MOV      (R5)+,(R2) ;RELOCATED RELOCF NOW CONTAINS RELOCATION
1297
1298 004352 000205          RTS      5          ;FACTOR
1299
1300
1301
1302          ;ROUTINE TO RELOCATE PROGRAM CODE FROM ORIGINAL POSITION (0-4K) TO
1303          ;TOP OF MEMORY.
1303 004354 012700 020000      RELOCF: MOV      #20000,R0 ;SET UP TO SCAN FOR TOP OF MEMORY
1304 004360 012737 000006 000004      MOV      #ERRVEC+2,#ERRVEC
1305 004366 062700 020000      1$:    ADD      #20000,R0 ;INCREMENT SCAN ADDRESS
1306 004372 000261          SEC
1307 004374 005710          TST      (R0)        ;SET TIME OUT INDICATOR
1308 004376 103373          BCC      1$          ;CHECK FOR EXISTANT MEMORY
1309 004400 012737 001126 000004      MOV      #ERRTRP,#ERRVEC ;'C' WILL BE CLEAR IF MEMORY EXISTS
1310 004406 162700 020000          SUB      #20000,R0 ;ADJUST TO LAST EXISTANT 4K
1311 004412 010067 000006          MOV      R0,2$      ;PASS RELOCATION ADDRESS TO RELOC ROUTINE
1312 004416 004567 177630          JSR      R5,RELOC   ;RELOCATE PROGRAM
1313 004422 000000          000000 ;FROM ADDRESS 000000
1314 004424 000000      2$:    .WORD      0 ;TO LAST 4K BANK
1315 004426 004567 174336          JSR      R5,$PRINT ;GO TO PRINT ROUTINE
1316 004432 010045          RELOCF
1317 004434 016746 177764          MOV      2$,-(SP)   ;PASS TO 02A ROUTINE
1318 004440 062716 010122          ADD      #REL24K,(SP) ;SET UP RESTART ADDRESS
1319 004444 004767 175344          JSR      PC,02A    ;TYPE RESTART ADDRESS
1320 004450 011667 000006          MOV      (SP),3$   ;SAVE RETURN ADDRESS IN 3$ BELOW
1321 004454 066706 177744          ADD      2$,SP     ;RESET STACK PTR
1322 004460 012716          MOV      (PC)+,(SP) ;GET RETURN ADDRESS
1323 004462 000000      3$:    .WORD      0 ;CONTAINS RETURN PC
1324 004464 066716 177734          ADD      2$,(SP)  ;ADJUST RETURN PC
1325 004470 000207          RTS      PC
1326
1327          .SBTTL MA/MF PARITY ERROR SERVICE ROUTINE
1328          ;WHEN MA/MF A PARITY ERROR IS DETECTED THIS ROUTINE SCANS MEMORY FOR THE
1329          ;ADDRESS CAUSING THE PARITY ERROR. WHEN THE ADDRESS IS LOCATED THE ROUTINE
1330          ;HALTS WITH THE ADDRESS+2 IN R0. TO CONTINUE AFTER THE ERROR PRESS CONTINUE.
1331 004472 010067 000170      .PARSRV: MOV      R0,SAVR0 ;SAVE R0 IN SAVRO
1332 004476 012700 004670          MOV      #SAVR0+2,R0

```



```

1333 004502 010120      MOV      R1,(R0)+
1334 004504 010220      MOV      R2,(R0)+
1335 004506 010320      MOV      R3,(R0)+
1336 004510 010420      MOV      R4,(R0)+
1337 004512 010520      MOV      R5,(R0)+
1338 004514 004567 174250      JSR      R5,$PRINT      ;GO TO PRINT ROUTINE
1339 004520 004702      PARERR
1340 004522 005027      CLR      (PC)+          ;CLEAR PARITY ERROR INDICATORS
1341 004524      000      PEFLG: .BYTE 0          ;NOT 0/0 =PAR ERR/NO PAR ERR
1342 004525      000      PENFLG: .BYTE 0      ;NOT 0/0=PAR ERR DETECTED/NOT DETECTED ON SCAN
1343 004526 012737 004574 000114      MOV      #2,$,2#PARVEC      ;SET PARITY ERROR TRAP
1344 004534 012737 004632 000004      MOV      #4,$,2#ERRVEC      ;SET TIME OUT TRAP VECTOR
1345 004542 005002      CLR      R2
1346 004544 005767 174212      TST      MMAVA          ;CHECK IF MEM MGMT IS AVAILABLE
1347 004550 001407      BEQ      1$            ;BRANCH IF NOT AVAILABLE
1348 004552 004767 000734      JSR      PC,LDMMO        ;SET UP MEM MGMT
1349 004556 105237 172301      INCB     2#KIPDR0+1      ;ALLOW FULL 4K PAGE ADDRESSING
1350 004562 012737 005614 000250      MOV      #MMABTO,2#MMVEC  ;SET MEM MGMT ABORT TRAP VECTOR
1351 004570 012200      1$:      MOV      (R2)+,R0      ;SCAN ALL ADDRESSES
1352 004572 000776      BR
1353 004574 110667 177724      2$:      MOV      SP,PEFLG      ;SET PARITY ERROR FOUND INDICATOR
1354 004600 010003      MOV      R0,R3
1355 004602 104400      HLT
1356 004604 000002      RTI
1357 004606 000240      3$:      NOP
1358 004610 005067 177710      CLR      PEFLG          ;PARITY ERROR! ADDRESS+2 IS IN R2
1359 004614 012706 000500      MOV      #STKPTR,SP      ;CONTINUE SCAN
1360 004620 000005      RESET      ;INSERT HALT INST TO EXAMINE PARITY REGS
1361 004622 004767 000504      JSR      PC,MAMF          ;CLEAR PARITY ERROR INDICATORS
1362 004626 000177 175622      JMP      2#PERSTRT       ;RESET STACK PTR
1363
1364      ;SERVICE ROUTINE IF PARITY ERROR NOT DETECTED ON SCAN
1365 004632 105767 177666      4$:      TSTB     PEFLG          ;BRANCH IF PARITY ERROR WAS
1366 004636 001363      BNE      3$            ;DETECTED ON SCAN
1367 004640 016602 000004      MOV      4(SP),R2        ;GET PC AT TIME OF ERROR
1368 004644 162702 000002      SUB      #2,R2           ;BACK IT UP
1369 004650 110667 177651      MOV      SP,PENFLG      ;SET IND = NO PAR ERROR DETECTED ON SCAN
1370 004654 004567 174110      JSR      R5,$PRINT      ;GO TO PRINT ROUTINE
1371 004660 004723      NOFIND
1372 004662 104400      HLT                    ;ERROR! PARITY ERROR NOT DETECTED ON SCAN
1373 004664 000750      BR      3$
1374
1375      ;THE BELOW 6 WORDS CONTAINS THE SAVED CONTENTS OF R0-R5 WHEN THE
1376 004666 000000      ;PARITY ERROR OCCURRED
1377 004670 000000      SAVR0: .WORD 0
1378 004672 000000      SAVR1: .WORD 0
1379 004674 000000      SAVR2: .WORD 0
1380 004676 000000      SAVR3: .WORD 0
1381 004700 000000      SAVR4: .WORD 0
1382
1383 004702 005015 040520 044522      PARERR: .ASCIZ <15><12>'PARITY ERROR'<15><12>
1384 004710 054524 042440 051122
1385 004716 051117 005015      000
1386 004723      116 052117 043040      NOFIND: .ASCIZ 'NOT FOUND ON SCAN'<15><12>
1387 004730 052517 042116 047440
1388 004736 020116 041523 047101
    
```


1389 004744 005015 000
 1390 004750 .EVEN
 1391
 1392
 1393 177740
 1394 177742
 1395 177744
 1396
 1397
 1398 004750 005767 177550
 1399 004754 001403
 1400 004756 000000
 1401 004760 000177 175470
 1402 004764 010667 177534
 1403 004770 005737 177570
 1404 004774 100001
 1405 004776 000000
 1406 005000 013746 177744
 1407 005004 013701 177740
 1408 005010 013702 177742
 1409 005014 011637 177744
 1410 005020 032737 020000 177570
 1411 005026 001066
 1412
 1413 005030 004567 173734
 1414 005034 004702
 1415
 1416 005036 004567 173726
 1417 005042 001475
 1418 005044 016646 000002
 1419 005050 066716 173710
 1420 005054 004767 174734
 1421 005060 004567 173704
 1422 005064 002361
 1423
 1424 005066 012700 002351
 1425 005072 012704 000010
 1426 005076 012705 000003
 1427 005102 005003
 1428 005104 006301
 1429 005106 106102
 1430 005110 006103
 1431 005112 077504
 1432 005114 116320 002312
 1433 005120 077412
 1434
 1435 005122 004567 173642
 1436 005126 002327
 1437
 1438 005130 004567 173634
 1439 005134 001531
 1440 005136 011605
 1441 005140 004767 174650
 1442 005144 004767 175211
 1443
 1444 005150 016700 173576

```

MEMLO=177740
MEMHI=177742
MEMERR=177744

.22PAR: TST      PEFLG      ;BEEN HERE BEFORE
        BEQ      1$        ;BRANCH IF NO
        HALT     ;YES -- DOUBLE PARITY ERROR

1$:     JMP      JPERSTR   ;SET PARITY ERROR FLAG
        MOV      SP,PEFLG ;HALT ON ERROR?
        TST      J#SWR    ;BRANCH IF NO
        BPL     100$      ;YES
        HALT     ;SAVE MEMORY ERROR REG
100$:   MOV      J#MEMERR,-(SP);GET ADDRESS OF WHERE THE PARITY
        MOV      J#MEMLO,R1;ERROR OCCURRED
        MOV      J#MEMHI,R2;CLEAR THE ERROR REG
        MOV      (SP),J#MEMERR;INHIBIT ERROR TYPEOUT
        BIT     #BIT13,J#SWR;BRANCH IF YES
        BNE     101$
        JSR     "PARITY ERROR"
        JSR     R5,$PRINT

;PRINT "PC=XXXXXX"
        JSR     R5,$PRINT
        JSR     ERRPC
        MOV     2(SP),-(SP) ;GET PC AT TIME OF PARITY ERROR
        ADD     RELOC, (SP)
        JSR     PC,02A
        JSR     R5,$PRINT
        SPACE1
;CHANGE 22-BIT ADDRESS TO OCTAL-ASCII
        MOV     #DIGITS,R0
        MOV     #8,R4
2$:     MOV     #3,R5
3$:     CLR     R3
4$:     ASL     R1
        ROLB   R2
        ROL    R3
        SOB   R5,4$
        MOVB  DIGTAB(R3),(R0)+
        SOB   R4,2$
;PRINT "MEMORY ADDRESS IS AAAAAAA"
        JSR     R5,$PRINT
        ADRESS
;PRINT "PARITY ERROR REG=XXXXXX"
        JSR     R5,$PRINT
        PARREG
        MOV     (SP),R5
        JSR     PC,02A
        JSR     PC,SPACE1
;PRINT THE MARGIN SETTING
        MOV     ICNT,R0

```



```

1445 005154 116000 005504      MOVB   MRGNTB(RO),RO
1446 005160 062700 005210      ADD    #MARTBL,RO
1447 005164 010067 000004      MOV    RO,5$
1448 005170 004567 173574      JSR    RS,$SPRINT
1449 005174 005210      5$:    MARTBL
1450 005176 004567 173566      JSR    RS,$SPRINT
1451 005202 005317      MARMMSG
1452 005204 000177 175244      101$:  JMP    @PERSTR
           :MARGIN MESSAGE TABLE
1453
1454 005210 005226      MARTBL: NORMAL
1455 005212 000000      0
1456 005214 005235      ESTRB
1457 005216 005252      LSTRB
1458 005220 005266      LCRNT
1459 005222 005302      HCRNT
1460 005224 005226      NORMAL
1461
1462      :MARGIN MESSAGES
1463 005226 047516 046522 046101  NORMAL: .ASCIZ 'NORMAL'
1464 005234      000
1465 005235      105 051101 054514  ESTRB: .ASCIZ 'EARLY STROBE'
1466 005242 051440 051124 041117
1467 005250 000105
1468 005252 040514 042524 051440  LSTRB: .ASCIZ 'LATE STROBE'
1469 005260 051124 041117 000105
1470 005266 047514 020127 052503  LCRNT: .ASCIZ 'LOW CURRENT'
1471 005274 051122 047105 000124
1472 005302 044510 044107 041440  HCRNT: .ASCIZ 'HIGH CURRENT'
1473 005310 051125 042522 052116
1474 005316      000
1475 005317      040 040515 043522  MARMMSG: .ASCIZ ' MARGIN'<12><15>
1476 005324 047111 006412      000
1477      005332      .EVEN
1478
1479      ;ROUTINE TO ENABLE PARITY ERROR ACTION ON MA/MF PARITY MEMORIES
1480      172100      PARCSR=172100      ;ADDRESS OF FIRST PARITY REGISTER
1481      000114      PARVEC=114      ;PARITY ERROR INTERRUPT VECTOR ADDRESS
1482
1483 005332 032737 000040 177570  .MAMF:  BIT    #40,@SWR      ;CHECK IF PARITY ERROR DETECTION IS TO
1484 005340 001033      BNE    DISPAR      ;BE ENABLED. BRANCH IF NOT TO BE ENABLED
1485 005342 013746 000004      MOV    @ERRVEC, -(SP) ;SAVE ERROR TRAP VECTOR
1486 005346 012737 000006 000004  MOV    #ERRVEC+2,@ERRVEC ;SET TIME OUT TRAP TO RETURN (VIA RTI)
1487 005354 012737 004472 000114  MOV    #.PARSRV,@PARVEC ;SET PARITY ERROR TRAP VECTOR
1488 005362 012737 000340 000116  MOV    #340,@PARVEC+2 ;PRIORITY LEVEL 7 ON TRAP
1489 005370 012700 172100      MOV    #PARCSR,RO   ;GET FIRST ADDRESS OF PARITY REGISTER
1490 005374 012702 000001      MOV    #1,R2
1491 005400 005027      CLR    (PC)+
1492 005402 000000      PARAVA: .WORD 0      ;CLEAR AVAILABILITY INDICATOR
           ;CONTAINS AVAILABILITY INDICATOR
1493
1494      ;ENABLE ALL AVAILABLE PARITY REGISTERS
1495 005404 000262      1$:    SEV
1496 005406 012720 000001      MOV    #1,(RO)+
           ;SET TIME OUT INDICATOR
           ;SET ACTION ENABLE IF AVAILABLE
1497 005412 102402      BVS    2$
           ;BRANCH IF NO PARITY AVAILABLE
1498 005414 050267 177762      BIS    R2,PARAVA
           ;SET AVAILABILITY INDICATOR
1499 005420 006302      2$:    ASL    R2
           ;SHIFT INDICATOR
1500 005422 103370      BCC    1$
    
```



```

1501 005424 012637 000004      MOV      (SP)+, @#ERRVEC ;RESTORE ERROR TRAP VECTOR
1502 005430 022767 177776 173324 DISPAR: CMP      #-2, MMAVA
1503 005436 001011      BNE      1$
1504 005440 012767 000001 177734      MOV      #1, PARAVA
1505 005446 012737 004750 000114      MOV      #.22PAR, @#PARVEC
1506 005454 012737 000340 000116      MOV      #340, @#PARVEC+2
1507 005462 000207      IS:     RTS      PC      ;RETURN
1508
1509      .SBTTL MARGIN ROUTINE
1510      :ROUTINE TO SET THE MARGINS
1511      CNTRL=177746
1512      177746      MAINTRG=177750
1513      177750
1514 005464 016700 173262      MARGIN: MOV      ICNT, R0      ;PASS COUNT
1515 005470 005002      CLR      R2      ;FAST COUNTER
1516 005472 116037 005504 177750      MOVVB   MRGNTB(R0), @#MAINTRG ;LOAD MAINTENANCE REG.
1517 005500 077201      IS:     SOB      R2, 1$
1518 005502 000207      RTS      PC
1519
1520 005504      000      MRGNTB: .BYTE   0      ;NORMAL
1521 005505      004      .BYTE   4      ;EARLY STROBE
1522 005506      006      .BYTE   6      ;LATE STROBE
1523 005507      010      .BYTE  10      ;LOW CURRENT
1524 005510      012      .BYTE  12      ;HIGH CURRENT
1525 005511      000      .BYTE   0      ;NORMAL
1526
1527
1528      .SBTTL MEM MGMT ROUTINES
1529      :ROUTINE TO INITIALIZE MEMORY MANAGEMENT REGISTERS
1530 005512 000240      LOMMO: NOP
1531 005514 005767 173242      TST      MMAVA
1532 005520 001434      BEQ      1$
1533 005522 012737 000020 172516      MOV      #20, @#SR3      ;22 BIT MODE
1534 005530 012737 077006 172300      MOV      #177*256.-400+UP+RW, @#KIPDR0 ;SET KIPDR0=RW UP 177 BLOCKS
1535 005536 012737 077406 172302      MOV      #200*256.-400+UP+RW, @#KIPDR1 ;SET KIPDR1=RW UP 200 BLOCKS
1536 005544 005037 172304      CLR      @#KIPDR2
1537 005550 005037 172344      CLR      @#KIPAR2
1538 005554 012737 077406 172316      MOV      #200*256.-400+UP+RW, @#KIPDR7 ;SET KIPDR7=RW UP 200 BLOCKS
1539 005562 005037 172340      CLR      @#KIPAR0
1540 005566 012737 000200 172342      MOV      #200, @#KIPAR1
1541 005574 012737 177600 172356      MOV      #177600, @#KIPAR7
1542 005602 012737 000001 177572      MOV      #1, @#SRO      ;ENABLE MEM MGMT
1543 005610 000240      NOP
1544 005612 000207      IS:     RTS      PC
1545
1546      :MEMORY MANAGEMENT ABORT ROUTINE FOR WRITE UP
1547 005614 012702 020000      MMABTO: MOV      #20000, R2      ;RESET R2
1548 005620 062737 000200 172342      ADD      #200, @#KIPAR1 ;ADVANCE TO NEXT 4K
1549 005626 013716 177576      MOV      @#SR2, (SP) ;RETURN TO INSTRUCTION THAT
1550 005632 005037 177572      CLR      @#SRO ;DISABLE MEM MGMT
1551 005636 012737 000001 177572      MOV      #1, @#SRO ;ENABLE MEM MGMT
1552 005644 000002      RTI ;CAUSED THE ABORT
1553
1554      :MEM MGMT ABORT SERVICE FOR WRITE DOWN
1555 005646 012702 040000      MMABT1: MOV      #40000, R2 ;RESET R2
1556 005652 162737 000200 172342      SUB      #200, @#KIPAR1
    
```



```

1557 005660 001406          BEQ      2$
1558 005662 013716 177576    MOV      @#SR2,(SP)
1559 005666 012737 000001 177572    MOV      #1,@#SR0          ;ENABLE MEM MGMT
1560 005674 000002          RTI
1561 005676          2$:
1562 005676 005037 177572          CLR      @#SR0          ;DISABLE MEM MGMT
1563 005702 052766 000002 000002    BIS      #V,2(SP)
1564 005710 000002          RTI
1565
1566          ;ROUTINE TO SET UP MEMORY MANAGEMENT FOR PATTERN TESTS
1567 005712 005702          STMM2:  TST      R2          ;CHECK IF TESTING BANK # 0
1568 005714 001442          BEQ      2$          ;EXIT IF BANK # 0
1569 005716 005767 173040    TST      MMAVA
1570 005722 001005          BNE      1$          ;BRANCH IF MEM MGMT AVAILABLE
1571 005724 006002          ROR      R2          ;ADJUST ADDRESS
1572 005726 006002          ROR      R2
1573 005730 006002          ROR      R2
1574 005732 006002          ROR      R2
1575 005734 000207          RTS      PC          ;RETURN
1576
1577 005736 004767 177550          1$:  JSR      PC,LDMMO          ;GO MAKE INITIAL SET UP
1578 005742 000302          SWAB    R2
1579 005744 006002          ROR      R2
1580 005746 010237 172344    MOV      R2,@#KIPAR2
1581 005752 062702 000200    ADD      #200,R2
1582 005756 010237 172346    MOV      R2,@#KIPAR3
1583 005762 012737 077406 172304    MOV      #200*256.-400+UP+RW,@#KIPDR2 ;SET KIPDR2=RW UP 200 BLOCKS
1584 005770 012737 077406 172306    MOV      #200*256.-400+UP+RW,@#KIPDR3 ;SET KIPDR3=RW UP 200 BLOCKS
1585 005776 005037 172310    CLR      @#KIPDR4
1586 006002 012702 040000    MOV      #40000,R2
1587 006006 012737 006024 000250    MOV      #MMABT2,@#MMVEC
1588 006014 012737 000001 177572    MOV      #1,@#SR0          ;ENABLE MEM MGMT
1589 006022 000207          2$:  RTS      PC
1590
1591          ;ROUTINE TO SERVICE 8 XOR 13 ABORTS
1592 006024 000240          MMABT2: NOP
1593 006026 012702 040000    MOV      #40000,R2
1594 006032 062737 000400 172344    ADD      #400,@#KIPAR2
1595 006040 062737 000400 172346    ADD      #400,@#KIPAR3
1596 006046 013716 177576    MOV      @#SR2,(SP)          ;SET RETURN TO INSTRUCTION THAT ABORTED
1597 006052 012737 000001 177572    MOV      #1,@#SR0          ;ENABLE MEM MGMT
1598 006060 000002          RTI
1599
1600          .SBTTL 3 XOR 9 ROUTINES
1601          ;ROUTINE TO WRITE 3XOR9 WORST CASE NOISE TEST PATTERN
1602          ;CALL:  MOV      BANK #,-(SP)          ;PUSH STARTING BANK # ON STACK
1603          ;      MOV      BLKCNT,-(SP)          ;PUSH 256. WORD BLOCK COUNT ON STACK
1604          ;      JSR      PC,.3X9          ;CALL ROUTINE
1605
1606 006062 016602 000004          .3X9:  MOV      4(SP),R2          ;GET STARTING BANK #
1607 006066 004767 177620          JSR      PC,STMM2
1608 006072 005000          CLR      R0
1609 006074 010003          MOV      R0,R3
1610 006076 005103          COM      R3          ;R0 (0) AND R3 (-1) IS THE DATA WRITTEN
1611 006100 005767 175352          TST      PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
1612 006104 001402          BEQ      1$          ;NOT TO BE WRITTEN
    
```


1669	006274	012703	177777	MOV	#-1,R3	
1670	006300	000207		RTS	PC	
1671						
1672						:ROUTINE TO CHECK 3 XOR 9 WORST CASE NOISE PATTERN
1673				:CALL:	MOV BANK#,-(SP)	:PUSH STARTING BANK # ONTO STACK
1674					MOV BLKCNT,-(SP)	:AND 256. WORD BLOCK COUNT
1675					JSR PC,..3X9	:CALL ROUTINE
1676						
1677	006302	000240		..3X9:	NOP	
1678	006304	004767	001104		JSR PC,CKSWR	:GO CHECK SWITCH REGISTER
1679						
1680						:CHECK WORST CASE PATTERN
1681	006310	016604	000002	1\$:	MOV 2(SP),R4	:GET 256. BLOCK WORD COUNT
1682	006314	016602	000004		MOV 4(SP),R2	:GET FIRST BANK #
1683	006320	004767	177366		JSR PC,STMM2	:GO SET UP MEM MGMT
1684	006324	005000			CLR R0	:SET CHECK WORD
1685	006326	005767	172422		TST ICOUNT	:IF ICOUNT IS NEG AM CHECKING COMP-
1686	006332	100001			BPL .+4	:LEMENTED PATTERN
1687	006334	005100			COM R0	:SO COMPLEMENT CHECK WORD
1688	006336	012705	000040	2\$:	MOV #32.,R5	:SET 256. WORD COUNTER
1689						
1690	006342	005767	175110	3\$:	TST PARPAT	:BRANCH IF PARITY MEMORY PATTERN IS
1691	006346	001402			BEQ 30\$:NOT TO BE CHECKED
1692						
1693	006350	004767	177640		JSR PC,.XOR39	:GO GET CONSTANT
1694	006354			30\$:		
1695	006354	012203			MOV (R2)+,R3	:GET TEST DATA
1696	006356	020003			CMP R0,R3	:COMPARE WITH CHECK WORD
1697	006360	001403			BEQ .+10	
1698	006362	005046			CLR -(SP)	:PUSH FAKE STATUS ON THE STACK
1699	006364	004767	172614		JSR PC,ERROR	:ERROR! MEM DATA (R3) NOT = TEST DATA
1700						: (R0), ADDRESS=(R2)-2
1701						
1702	006370	012203			MOV (R2)+,R3	:GET TEST DATA
1703	006372	020003			CMP R0,R3	:COMPARE WITH CHECK WORD
1704	006374	001403			BEQ .+10	
1705	006376	005046			CLR -(SP)	:PUSH FAKE STATUS ON THE STACK
1706	006400	004767	172600		JSR PC,ERROR	:ERROR! MEM DATA (R3) NOT = TEST DATA
1707						: (R0), ADDRESS=(R2)-2
1708						
1709	006404	012203			MOV (R2)+,R3	:GET TEST DATA
1710	006406	020003			CMP R0,R3	:COMPARE WITH CHECK WORD
1711	006410	001403			BEQ .+10	
1712	006412	005046			CLR -(SP)	:PUSH FAKE STATUS ON THE STACK
1713	006414	004767	172564		JSR PC,ERROR	:ERROR! MEM DATA (R3) NOT = TEST DATA
1714						: (R0), ADDRESS=(R2)-2
1715						
1716	006420	012203			MOV (R2)+,R3	:GET TEST DATA
1717	006422	020003			CMP R0,R3	:COMPARE WITH CHECK WORD
1718	006424	001403			BEQ .+10	
1719	006426	005046			CLR -(SP)	:PUSH FAKE STATUS ON THE STACK
1720	006430	004767	172550		JSR PC,ERROR	:ERROR! MEM DATA (R3) NOT = TEST DATA
1721						: (R0), ADDRESS=(R2)-2
1722						
1723	006434	012203			MOV (R2)+,R3	:GET TEST DATA
1724	006436	020003			CMP R0,R3	:COMPARE WITH CHECK WORD


```

1725 006440 001403          BEQ      .+10
1726 006442 005046          CLR      -(SP)
1727 006444 004767 172534   JSR      PC,ERROR          ;PUSH FAKE STATUS ON THE STACK
                                ;ERROR! MEM DATA (R3) NOT = TEST DATA
                                ;(R0), ADDRESS=(R2)-2
1728
1729
1730 006450 012203          MOV      (R2)+,R3          ;GET TEST DATA
1731 006452 020003          CMP      R0,R3            ;COMPARE WITH CHECK WORD
1732 006454 001403          BEQ      .+10
1733 006456 005046          CLR      -(SP)
1734 006460 004767 172520   JSR      PC,ERROR          ;PUSH FAKE STATUS ON THE STACK
                                ;ERROR! MEM DATA (R3) NOT = TEST DATA
                                ;(R0), ADDRESS=(R2)-2
1735
1736
1737 006464 012203          MOV      (R2)+,R3          ;GET TEST DATA
1738 006466 020003          CMP      R0,R3            ;COMPARE WITH CHECK WORD
1739 006470 001403          BEQ      .+10
1740 006472 005046          CLR      -(SP)
1741 006474 004767 172504   JSR      PC,ERROR          ;PUSH FAKE STATUS ON THE STACK
                                ;ERROR! MEM DATA (R3) NOT = TEST DATA
                                ;(R0), ADDRESS=(R2)-2
1742
1743
1744 006500 012203          MOV      (R2)+,R3          ;GET TEST DATA
1745 006502 020003          CMP      R0,R3            ;COMPARE WITH CHECK WORD
1746 006504 001403          BEQ      .+10
1747 006506 005046          CLR      -(SP)
1748 006510 004767 172470   JSR      PC,ERROR          ;PUSH FAKE STATUS ON THE STACK
                                ;ERROR! MEM DATA (R3) NOT = TEST DATA
                                ;(R0), ADDRESS=(R2)-2
1749
1750
1751
1752 006514 005100          COM      R0                ;COMPLEMENT CHECK WORD
1753 006516 005305          DEC      R5                ;DECREMENT 256. WORD COUNTER
1754 006520 001310          BNE      3$
1755 006522 005100          COM      R0                ;COMPLEMENT CHECK WORD
1756 006524 005304          DEC      R4                ;DECREMENT BLOCK COUNTER
1757 006526 001303          BNE      2$
1758
1759 006530 032737 040000 177570 BIT      #40000,2$SWR      ;LOOP ON TEST?
1760 006536 001264          BNE      1$                ;BRANCH IF LOOP ON TEST DESIRED
1761 006540 016667 000002 172220 40$: MOV      2(SP),COUNT      ;GET # OF 256. WORD BLOCKS TO CHECK
1762 006546 016602 000004          MOV      4(SP),R2          ;GET STARTING BANK #
1763 006552 004767 177134          JSR      PC,STMM2          ;GO SET UP MEM MGMT IF REQUIRD
1764
1765          ;CHECK WORST CASE BIT COMPLEMENT PATTERN
1766 006556 005000          CLR      R0
1767 006560 005767 172170   TST      ICOUNT            ;CHECK IF COMPLEMENT PATERN
1768 006564 100001          BPL      .+4
1769 006566 005100          COM      R0                ;COMPLEMENT CHECK WORD
1770 006570 012704 000040 4$: MOV      #32.,R4          ;SET 256. WORD COUNTER
1771 006574 012705 000010 5$: MOV      #8.,R5           ;SET 8 WORD COUNTER
1772 006600 005767 174652 6$: TST      PARPAT          ;BRANCH IF PARITY MEMORY PATTERN IS
                                ;NOT TO BE CHECKED
1773 006604 001402          BEQ      60$
1774 006606 004767 177402   JSR      PC,.XOR39
1775 006612 012203 60$: MOV      (R2)+,R3          ;GET DATA
1776 006614 020003          CMP      R0,R3            ;CHECK DATA
1777 006616 001403          BEQ      .+10
1778 006620 005046          CLR      -(SP)
1779 006622 004767 172356   JSR      PC,ERROR
1780 006626 005100          COM      R0                ;COMPLEMENT CHECK WORD

```


1781	006630	005142		COM	-(R2)	;COMPLEMENT TEST DATA
1782	006632	012203		MOV	(R2)+,R3	;GET DATA
1783	006634	020003		CMP	RO,R3	;CHECK
1784	006636	001403		BEQ	+.10	
1785	006640	005046		CLR	-(SP)	;PUSH FAKE STATUS ON THE STACK
1786	006642	004767	172336	JSR	PC,ERROR	
1787	006646	005100		COM	RO	;COMPLEMENT CHECK WORD
1788	006650	005162	177776	COM	-2(R2)	;RESTORE DATA
1789	006654	005305		DEC	R5	;DECREMENT 4 WORD COUNTER
1790	006656	001350		BNE	6\$	
1791	006660	005100		COM	RO	;COMPLEMENT CHECK WORD
1792	006662	005304		DEC	R4	;DECREMENT 256. WORD COUNTER
1793	006664	001343		BNE	5\$	
1794	006666	005100		COM	RO	;COMPLEMENT CHECK WORD
1795	006670	005367	172072	DEC	COUNT	;DECREMENT BLOCK COUNTER
1796	006674	001335		BNE	4\$	
1797						
1798	006676	016602	000004	MOV	4(SP),R2	;GET BANK #
1799	006702	004767	177004	JSR	PC,STMM2	
1800	006706	016603	000002	MOV	2(SP),R3	;GET BLOCK COUNT
1801	006712	032737	040000	BIT	#40000,2#SWR	;LOOP ON TEST
1802	006720	001307		BNE	40\$;BRANCH IF LOOP ON TEST
1803	006722	006367	172026	ASL	ICOUNT	
1804	006726	102402		BVS	7\$	
1805	006730	000167	177354	JMP	1\$	
1806	006734	012705	000020	MOV	#16.,R5	;COMPLEMENT PATTERN
1807	006740	011200		MOV	(R2),RO	;GET 1ST DATA WORD
1808	006742	016204	000020	MOV	20(R2),R4	;GET 9TH DATA WORD
1809	006746	110422		MOVB	R4,(R2)+	;SWAP WORDS 1-8
1810	006750	110422		MOVB	R4,(R2)+	;WITH 9-16
1811	006752	110422		MOVB	R4,(R2)+	
1812	006754	110422		MOVB	R4,(R2)+	
1813	006756	110422		MOVB	R4,(R2)+	
1814	006760	110422		MOVB	R4,(R2)+	
1815	006762	110422		MOVB	R4,(R2)+	
1816	006764	110422		MOVB	R4,(R2)+	
1817	006766	110422		MOVB	R4,(R2)+	
1818	006770	110422		MOVB	R4,(R2)+	
1819	006772	110422		MOVB	R4,(R2)+	
1820	006774	110422		MOVB	R4,(R2)+	
1821	006776	110422		MOVB	R4,(R2)+	
1822	007000	110422		MOVB	R4,(R2)+	
1823	007002	110422		MOVB	R4,(R2)+	
1824	007004	110422		MOVB	R4,(R2)+	
1825	007006	110022		MOVB	RO,(R2)+	;AND VICE VERSA
1826	007010	110022		MOVB	RO,(R2)+	
1827	007012	110022		MOVB	RO,(R2)+	
1828	007014	110022		MOVB	RO,(R2)+	
1829	007016	110022		MOVB	RO,(R2)+	
1830	007020	110022		MOVB	RO,(R2)+	
1831	007022	110022		MOVB	RO,(R2)+	
1832	007024	110022		MOVB	RO,(R2)+	
1833	007026	110022		MOVB	RO,(R2)+	
1834	007030	110022		MOVB	RO,(R2)+	
1835	007032	110022		MOVB	RO,(R2)+	
1836	007034	110022		MOVB	RO,(R2)+	

7\$:
10\$:


```

1837 007036 110022          MOVB   R0,(R2)+
1838 007040 110022          MOVB   R0,(R2)+
1839 007042 110022          MOVB   R0,(R2)+
1840 007044 110022          MOVB   R0,(R2)+
1841 007046 005305          DEC    R5
1842 007050 001333          BNE   10$
1843 007052 005303          DEC    R3
1844 007054 001327          BNE   7$
1845
1846 007056 005767 171672      TST    ICOUNT
1847 007062 001402          BEQ   11$
1848 007064 000167 177220      JMP
1849 007070 012616          11$:  MOV   (SP)+,(SP)
1850 007072 012616          MOV   (SP)+,(SP)
1851 007074 000207          RTS   PC
1852
1853          ;ROUTINE TO WRITE 8 XOR 13 WORST CASE NOISE TEST PATTERN
1854          .SBTTL 8 XOR 13 ROUTINES
1855          ;CALL: MOV   BANK #,-(SP)
1856          ;      MOV   #4KBANKS,-(SP)
1857          ;      JSR   PC,.8X13
1858
1859 007076 012616          .8X13: MOV  (SP)+,(SP)      ;ADJUST STACK
1860 007100 012616          MOV  (SP)+,(SP)
1861 007102 000207          RTS   PC
1862
1863          ;ROUTINE TO CHECK 8 XOR 13 WORST CASE NOISE TEST PATTERN
1864          ;CALL:
1865          ;      MOV   BANK #,-(SP)      ;PUSH FIRST BANK # ON THE STACK
1866          ;      MOV   #BANKS,-(SP)     ;PUSH # OF 4K BANKS TO CHECK ON THE STACK
1867          ;      JSR   PC,..8X13       ;CALL ROUTINE
1868
1869 007104 012616          ..8X13: MOV (SP)+,(SP)
1870 007106 012616          MOV  (SP)+,(SP)
1871 007110 000207          RTS   PC      ;RETURN
1872
1873          .SBTTL ROTATING 1'S & 0'S ROUTINES
1874          ;ROUTINE TO CHECK ROTATING '0' BIT THROUGH FIELD OF 1'S
1875          ;CALL: MOV   BANK #,-(SP)      ;SET STARTING BANK #
1876          ;      MOV   BLKCNT,-(SP)     ;SET 256. WORD BLOCK COUNT
1877          ;      JSR   PC,.ROTD        ;CALL ROUTINE
1878
1879 007112 004767 000276      .ROTD: JSR   PC,CKSWR      ;GO CHECK SWITCHES
1880 007116 016604 000002      MOV   2(SP),R4          ;GET 256. WORD BLOCK COUNT
1881 007122 016602 000004      MOV   4(SP),R2          ;GET FIRST BANK #
1882 007126 004767 176560      JSR   PC,STMM2         ;GO SET UP MEM MGMT (IF AVAIL)
1883 007132 012700 177777      MOV   #-1,R0           ;SET CHECK WORD
1884
1885 007136 012705 000400      1$:   MOV   #256.,R5      ;SET 256. WORD COUNT
1886 007142 000241 2$:   CLC                    ;CLEAR CARRY BIT IN PSW
1887 007144 004767 000124      JSR   PC,ROTATE
1888 007150 016203 177776      MOV   -2(R2),R3        ;GET RESULT
1889 007154 103402          BCS   3$               ;BRANCH IF 'C' BIT WAS SET
1890 007156 020003          CMP   R0,R3            ;CHECK RESULT
1891 007160 001403          BEQ   4$
1892 007162 005046          3$:   CLR   -(SP)          ;ERROR! COULD NOT ROTATE '0' BIT
    
```



```

1893 007164 004767 172014          JSR    PC,ERROR          ;THROUGH ADDRESS IN R2
1894 007170 005305          4$:  DEC    R5           ;DECREMENT 256. WORD COUNT
1895 007172 001363          BNE    2$             ;LOOP UNTIL DONE
1896 007174 005304          DEC    R4           ;DECREMENT 256. WORD BLOCK COUNT
1897 007176 001357          BNE    1$             ;LOOP UNTIL DONE
1898 007200 012616          MOV    (SP)+,(SP)    ;POP CONSTANTS OFF THE STACK
1899 007202 012616          MOV    (SP)+,(SP)
1900 007204 000207          RTS    PC           ;RETURN TO CALLER
1901
1902          ;ROUTINE TO CHECK ROTATING '1' BIT THROUGH A FIELD OF 0'S
1903          ;CALL: MOV    BANK#,-(SP) ;SET STARTING BANK #
1904          ;      MOV    BLKCNT,-(SP) ;SET # OF 256. WORD BLOCKS TO CHECK
1905          ;      JSR    PC,.ROT1   ;CALL ROUTINE
1906
1907 007206 004767 000202      .ROT1: JSR    PC,CKSWR    ;GO CHECK SWITCHES
1908 007212 016604 000002      MOV    2(SP),R4     ;GET # OF 256. WORD BLOCKS TO CHECK
1909 007216 016602 000004      MOV    4(SP),R2     ;GET STARTING BANK #
1910 007222 004767 176464      JSR    PC,STMM2     ;GO SET UP MEM MGMT (IF AVAIL)
1911 007226 005000          CLR    R0           ;SET CHECK WORD
1912
1913 007230 012705 000400      1$:  MOV    #256.,R5   ;SET 256. WORD COUNTER
1914 007234 000261          2$:  SEC           ;SET 'C' BIT IN PSW
1915 007236 004767 000032      JSR    PC,ROTATE    ;GO ROTATE '1' BIT
1916 007242 016203 177776      MOV    -2(R2),R3    ;GET RESULT
1917 007246 103002          BCC    3$           ;BRANCH IF 'C' IS CLEAR
1918 007250 020003          CMP    R0,R3        ;CHECK RESULT
1919 007252 001401          BEQ    .+4
1920 007254 104400          3$:  HLT           ;ERROR! COULD NOT ROTATE '1' BIT
1921
1922 007256 005305          DEC    R5           ;THROUGH ADDRESS IN R2
1923 007260 001365          BNE    2$           ;DECREMENT 256. WORD COUNT
1924 007262 005304          DEC    R4           ;DECREMENT 256. WORD BLOCK COUNT
1925 007264 001361          BNE    1$           ;ADJUST RETURN ADDRESS
1926 007266 012616          MOV    (SP)+,(SP)
1927 007270 012616          MOV    (SP)+,(SP)
1928 007272 000207          RTS    PC           ;RETURN TO CALLER
1929
1930          ;ROUTINE TO ROTATE 'C' BIT THROUGH A MEMORY LOCATION.
1931 007274 106112      ROTATE: ROLB    (R2)    ;(R2)=177776 OR 000001
1932 007276 106112      ROLB    (R2)    ;(R2)=177775 OR 000002
1933 007300 106112      ROLB    (R2)    ;(R2)=177773 OR 000004
1934 007302 106112      ROLB    (R2)    ;(R2)=177767 OR 000010
1935 007304 106112      ROLB    (R2)    ;(R2)=177757 OR 000020
1936 007306 106112      ROLB    (R2)    ;(R2)=177737 OR 000040
1937 007310 106112      ROLB    (R2)    ;(R2)=177677 OR 000100
1938 007312 106112      ROLB    (R2)    ;(R2)=177777 OR 000000
1939 007314 106122      ROLB    (R2)+    ;(R2)=177577 OR 000200
1940 007316 106112      ROLB    (R2)    ;(R2)=177377 OR 000400
1941 007320 106112      ROLB    (R2)    ;(R2)=176777 OR 001000
1942 007322 106112      ROLB    (R2)    ;(R2)=175777 OR 002000
1943 007324 106112      ROLB    (R2)    ;(R2)=173777 OR 004000
1944 007326 106112      ROLB    (R2)    ;(R2)=167777 OR 010000
1945 007330 106112      ROLB    (R2)    ;(R2)=157777 OR 020000
1946 007332 106112      ROLB    (R2)    ;(R2)=137777 OR 040000
1947 007334 106112      ROLB    (R2)    ;(R2)=077777 OR 100000
1948 007336 106122      ROLB    (R2)+    ;(R2)=177777 OR 000000
    
```



```

1949 007340 000207          RTS      PC          ;RETURN
1950
1951          ;ROUTINE TO WRITE ONE WORD PATTERN INTO MEMORY
1952          ;CALL:  MOV      BANK#,-(SP)      ;PUSH STARTING BANK # ONTO STACK
1953          ;      MOV      BLKCNT,-(SP)      ;AND 128. WORD BLOCK COUNT
1954          ;      JSR      PC,WRTPAT        ;CALL ROUTINE
1955
1956 007342 016604 000002    WRTPAT: MOV      2(SP),R4      ;GET BLOCK COUNT
1957 007346 016602 000004    MOV      4(SP),R2      ;GET STARTING BANK #
1958 007352 004767 176334    JSR      PC,STMM2     ;GO SET UP MEM MGMT
1959 007356 012700          MOV      (PC)+,R0     ;GET USER CONSTANT
1960 007360 000000          .CONST: 0
1961 007362 012703 000100    1$:  MOV      #64,R3      ;SET 256. WORD COUNTER
1962 007366 010022          2$:  MOV      R0,(R2)+   ;WRITE 256. WORDS
1963 007370 010022          MOV      R0,(R2)+
1964 007372 010022          MOV      R0,(R2)+
1965 007374 010022          MOV      R0,(R2)+
1966 007376 005303          DEC      R3          ;DECREMENT 256. WORD COUNTER
1967 007400 001372          BNE     2$          ;LOOP UNTIL 256. WORDS HAVE BEEN WRITTEN
1968 007402 005304          DEC      R4          ;DECREMENT BLOCK COUNT
1969 007404 001366          BNE     1$
1970 007406 012616          MOV      (SP)+,(SP)  ;ADJUST STACK
1971 007410 012616          MOV      (SP)+,(SP)
1972 007412 000207          RTS      PC
1973
1974
1975          ;ROUTINE TO CHECK THE SWITCH REGISTER
1976          ;CHECK SWITCH 9: IF SET, LOAD ERROR COUNT INTO THE DISPLAY REGISTER;
1977          ;IF NOT SET, LOAD PASS COUNT INTO THE DISPLAY REGISTER.
1978 007414 042767 017777 171336  CKSWR: BIC      #17777,LDISP  ;SAVE RELOCATION BITS
1979 007422 032737 000400 177570    BIT      #BIT8,#SWR    ;CHECK SWITCH 8
1980 007430 001402          BEQ     10$          ;BRANCH IF SET
1981 007432 004767 000464          JSR      PC,REL24K    ;GO RELOCATE PROGRAM BACK TO 4K AND STOP
1982 007436 032737 001000 177570  10$:  BIT      #BIT9,#SWR  ;SWITCH 9 SET ?
1983 007444 001404          BEQ     1$
1984 007446 056767 171304 171304    BIS      ERcnt,LDISP  ;LOAD ERROR COUNT
1985 007454 000403          BR      2$
1986 007456 056767 171270 171274  1$:  BIS      ICNT,LDISP   ;LOAD PASS COUNT
1987 007464 016737 171270 177570  2$:  MOV      LDISP,#DISPLAY ;LOAD THE DISPLAY REGISTER
1988 007472 012767 040177 171254    MOV      #040177,ICOUNT ;LOAD ITERATION COUNT WORD
1989 007500 032737 004000 177570    BIT      #4000,#SWR   ;CHECK SW11
1990 007506 001402          BEQ     +6
1991 007510 105067 171240          CLRB   ICOUNT        ;ICOUNT =040000 IF SW11 =1
1992 007514 000207          RTS      PC
1993
1994          ;MESSAGES
1995 007516 005015 047524 051040  RESLDR: .ASCIZ <15><12>'TO RESTORE LOADERS START AT 162'<15><12>
1996 007524 051505 047524 042522
1997 007532 046040 040517 042504
1998 007540 051522 051440 040524
1999 007546 052122 040440 020124
2000 007554 033061 006462 000012
2001 007562 005015 047105 041101  PARITY: .ASCIZ <15><12>'ENABLE PARITY? 1/0=YES/NO '
2002 007570 042514 050040 051101
2003 007576 052111 037531 030440
2004 007604 030057 054475 051505

```



```

2005 007612 047057 020117 000
2006 007617 015 051412 040524 STBANK: .ASCIZ <15><12>'STARTING BANK #(8)? '
2007 007624 052122 047111 020107
2008 007632 040502 045516 021440
2009 007640 034050 037451 000040
2010 007646 005015 020043 043117 BANKS: .ASCIZ <15><12>'# OF 4K BANKS TO TEST(8)? '
2011 007654 032040 020113 040502
2012 007662 045516 020123 047524
2013 007670 052040 051505 024124
2014 007676 024470 020077 000
2015 007703 015 050012 052101 PAT: .ASCIZ <15><12>'PATTERN #' '
2016 007710 042524 047122 021440
2017 007716 020077 000
2018 007721 015 037412 000 QUEST: .ASCIZ <15><12>'?'
2019 007725 015 052012 050131 CONST: .ASCIZ <15><12>'TYPE CONSTANT'
2020 007732 020105 047503 051516
2021 007740 040524 052116 000
2022 007745 015 044412 050116 PRG3M: .ASCIZ <15><12>'INPUT # OF 256. WORD BLOCKS TO TEST INSTEAD OF'
2023 007752 052125 021440 047440
2024 007760 020106 032462 027066
2025 007766 053440 051117 020104
2026 007774 046102 041517 051513
2027 010002 052040 020117 042524
2028 010010 052123 044440 051516
2029 010016 042524 042101 047440
2030 010024 000106
2031 010026 005015 054524 042520 PRG4M: .ASCIZ <15><12>'TYPE ADDRESS'
2032 010034 040440 042104 042522
2033 010042 051523 000
2034 010045 015 052012 020117 RELOCM: .ASCIZ <15><12>'TO RESTORE PROGRAM START AT '
2035 010052 042522 052123 051117
2036 010060 020105 051120 043517
2037 010066 040522 020115 052123
2038 010074 051101 020124 052101
2039 010102 000040
2040 010104 000052
2041 010106 042504 045115 020101 ASTERISK: .ASCIZ '*'
2042 010114 047504 042516 000041 ENDMSG: .ASCIZ 'DEMJA DONE!'
2043 .EVEN
2044
2045 ;ROUTINE TO RELOCATE PROGRAM BACK TO 0
2046 010122 010700 REL24K: MOV PC,RO ;FORM BASE ADDRESS WHERE CODE
2047 010124 042700 017777 BIC #17777,RO ;IS RELOCATED
2048 010130 010067 000004 MOV RO,1$ ;PUT FROM ADDRESS INTO SUBROUTINE CALL
2049 010134 004567 174112 JSR R5,RELOC ;RELOCATE CODE TO
2050 010140 000000 1$: 0 ;LOWEST 4K
2051 010142 000000 0
2052 010144 012706 000500 MOV #STKPTR,SP ;SET STACK PTR
2053 010150 042737 100000 000760 BIC #100000,@#LDDISP ;CLEAR RELOCATION INDICATOR
2054 010156 013737 000760 177570 MOV @#LDDISP,@#DISPLAY ;LOAD DISPLAY REGISTER
2055 010164 005037 000764 CLR @#RELOCF ;CLEAR RELOCATION FACTOR
2056 010170 000005 RESET ;DISABLE MEM MGMT
2057 010172 000137 000162 JMP @#PONE ;RESTORE LOADERS & HALT
2058
2059 010176 LODAR:
2060 000001 .END

```


PC	=%000007	342*	471*	476*	496*	508*	518*	532*	555*	557*	560*	595*	622*	630*
		636*	643	648*	676*	678*	684*	694*	699*	700*	701*	730*	773*	776*
		791*	810*	812*	819*	836*	841*	849	854*	898*	906*	924*	930	931*
		940*	943*	946	951*	973	1003	1023	1024*	1039	1040*	1056	1071	1072*
		1112*	1113*	1114*	1118*	1127	1131*	1137*	1140	1143*	1158	1161*	1165	1172*
		1180	1184*	1221*	1222	1225*	1232*	1233	1236*	1240	1243	1257*	1258*	1319*
		1322	1325*	1340*	1348*	1361*	1420*	1441*	1442*	1491*	1507*	1518*	1544*	1575*
		1577*	1589*	1607*	1644*	1649*	1667*	1670*	1678*	1683*	1693*	1699*	1706*	1713*
		1720*	1727*	1734*	1741*	1748*	1763*	1774*	1779*	1786*	1799*	1851*	1861*	1871*
		1879*	1882*	1887*	1893*	1900*	1907*	1910*	1915*	1928*	1949*	1958*	1959	1972*
		1981*	1992*	2046										
PDWN	000502	517*	559	910										
PEFLG	004524	689	905*	1341*	1353*	1358*	1365	1398	1402*					
PENFLG	004525	687	1342*	1369*										
PERSTR	002454	907*	1362	1401	1452									
PFSTK	000564	533*	539											
PFVEC =	000024	373*	534*	559*	910*	911*								
PIRQ =	177772	385*												
PIRVEC=	000240	378*												
PKM =	000000	361*												
PLACE	002374	881*	1196											
PONE	000162	475*	2057											
PRG3M	007745	2022*												
PRG4M	010026	2031*												
PRTY4 =	000200	357*												
PRTY7 =	000340	356*	462											
PSM =	010000	362*												
PSW =	177776	383*	594											
PTWO	000200	478	480*											
PUM =	030000	363*												
PJP	000576	534	538*											
PWRFAI	000730	565	569*											
QUEST	007721	2018*												
RANTST	003702	1180*												
RECDAT	001514	697	719*											
REG =	004000	364*												
RELF	000110	472*												
RELOC	004252	1267*	1312	2049										
RELOCF	000764	585*	593	728	729	774	775	909*	1294	1419	2055*			
RELOCM	010045	1316	2034*											
RELOCP	004354	1303*												
REL24K	010122	1318	1981	2046*										
RESLDR	007516	900	1995*											
RESVEC=	000010	368*												
ROTATE	007274	1887	1915	1931*										
ROTO	004054	1218*												
ROT1	004114	1229*												
RW =	000006	436*	1186	1534	1535	1538	1583	1584						
RO	=%000000	335*	495	502*	524*	527	529*	530	542*	545*	547*	548*	612	613*
		616	619*	693	709	747*	750*	755*	759*	761*	771*	792*	795*	802*
		804*	806*	820*	822	825	827*	828	831	949*	950*	952*	954	961
		1010*	1011*	1012*	1013*	1015	1027*	1028*	1030	1042*	1043*	1046	1059*	1060*
		1062	1079	1086	1091*	1094	1135*	1136*	1137	1142	1197*	1199	1255*	1258
		1267*	1274	1280	1303*	1305*	1307	1310*	1311	1331	1332*	1333*	1334*	1335*
		1336*	1337*	1351*	1354	1424*	1432*	1444*	1445*	1446*	1447	1489*	1496*	1514*
		1516	1608*	1609	1614*	1617	1618	1619	1620	1622	1623	1624	1625	1639*

L04

MAINDEC-11-DEMJA-B PDP11/70 MEMORY TEST MACY11 27(732) 09-SEP-76 17:07 PAGE 49
DEMJAB.P11 CROSS REFERENCE TABLE -- MACRO NAMES

STYPE 311*

ADC	753														
ADD	593	614	664	710	728	729	752	797	936	952	989	995	1012	1212	1270
	1295	1305	1318	1321	1324	1419	1446	1548	1581	1594	1595				
ASL	748	763	923	981	982	983	984	985	986	1428	1499	1803			
BCC	1308	1500	1917												
BCS	1889														
BEQ	520	541	554	663	666	703	737	739	818	826	955	962	979	1005	1016
	1047	1065	1077	1080	1087	1095	1103	1147	1149	1170	1183	1200	1208	1246	1256
	1279	1281	1283	1347	1399	1532	1557	1568	1612	1642	1654	1656	1659	1691	1697
	1704	1711	1718	1725	1732	1739	1746	1773	1777	1784	1847	1891	1919	1980	1983
	1990														
BIC	667	740	751	774	775	988	1169	1978	2047	2053					
BIS	669	1498	1563	1984	1986										
BIT	665	674	702	738	915	1410	1483	1653	1655	1658	1759	1801	1979	1982	1989
BLT	629														
BMI	1662														
BNE	617	624	645	675	688	690	760	767	772	809	833	840	848	916	922
	999	1032	1050	1067	1090	1098	1117	1151	1194	1205	1276	1368	1411	1484	1503
	1570	1638	1646	1754	1757	1760	1790	1793	1796	1802	1842	1844	1895	1897	1923
	1925	1967	1969												
BPL	563	634	641	671	707	1105	1254	1404	1661	1664	1686	1768			
BR	478	631	673	762	823	937	957	971	991	1018	1033	1051	1068	1082	1099
	1153	1211	1213	1287	1290	1352	1373	1657	1985						
BVS	997	1063	1497	1804											
CLC	1986														
CLR	560	655	732	754	770	794	801	853	901	902	905	908	909	911	914
	918	1027	1042	1059	1114	1130	1160	1189	1229	1272	1340	1345	1358	1427	1491
	1515	1536	1537	1539	1550	1562	1585	1608	1684	1698	1705	1712	1719	1726	1733
	1740	1747	1766	1778	1785	1892	1911	2055							
CLRB	1991														
CMP	662	824	825	921	954	961	998	1015	1035	1046	1052	1064	1075	1079	1086
	1094	1101	1116	1146	1199	1243	1275	1278	1280	1291	1502	1696	1703	1710	1717
	1724	1731	1738	1745	1776	1783	1890	1918							
CMPB	623														
COM	919	958	959	994	1011	1083	1084	1292	1610	1639	1640	1687	1752	1755	1769
	1780	1781	1787	1788	1791	1794									
DEC	759	766	771	808	832	835	847	1031	1049	1060	1066	1097	1193	1204	1637
	1645	1753	1756	1789	1792	1795	1841	1843	1894	1896	1922	1924	1966	1968	
DECB	628														
EMT	433														
HALT	439	477	535	642	653	672	708	1286	1288	1400	1405				
INC	562	1028	1043	1089	1091	1209	1210	1242							
INCB	1185	1349													
JMP	481	656	1249	1262	1362	1401	1452	1805	1848	2057					
JSR	469	476	518	555	557	564	595	622	630	648	649	651	676	678	679
	681	684	685	691	694	696	699	700	701	704	730	773	791	799	810
	851	898	899	924	931	940	941	943	951	1024	1040	1072	1112	1113	1118
	1131	1143	1161	1172	1184	1221	1225	1232	1236	1247	1251	1257	1258	1312	1315
	1319	1338	1348	1361	1370	1413	1416	1420	1421	1435	1438	1441	1442	1448	1450
	1577	1607	1644	1678	1683	1693	1699	1706	1713	1720	1727	1734	1741	1748	1763
	1774	1779	1786	1799	1879	1882	1887	1893	1907	1910	1915	1958	1981	2049	
MOV	475	480	489	490	491	492	493	494	495	496	501	502	503	504	505
	506	507	508	517	521	522	523	524	525	526	527	529	530	532	534
	539	542	543	544	545	547	548	550	551	552	558	559	592	594	612
	613	619	625	646	647	669	677	683	693	698	709	731	734	735	746
	747	755	761	768	792	793	798	811	820	821	828	829	830	831	834

.SSTL	313	511	575	658	882	1107	1178	1264	1265	1327	1509	1528	1600	1854	1873
.TITLE	312														
.WORD	455	456	458	459	461	462	472	473	483	484	533	561	577	578	585
	586	596	610	611	644	813	815	850	856	881	907	1115	1138	1244	1245
	1314	1323	1376	1377	1378	1379	1380	1381	1492						

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*DEMJAB.DEMJAB.SEG/SOL/CRF/DS:ERFZ/EN:ABS=DSKM:DEMJAB.P11
 RUN-TIME: 14 12 3 SECONDS
 RUN-TIME RATIO: 53/31=1.7
 CORE USED: 8K (15 PAGES)

