

MS/MF/MA11

COMBINE MS11,MF11,MA11-P
MD-11-DCMFA-D

EP-DCMFA-D-DL-A

APR 1977

COPYRIGHT © 1977

digital

FICHE 1 OF 1

MADE IN USA

This microfiche card contains a grid of frames. The frames are arranged in approximately 10 rows and 6 columns. Each frame contains a small, high-contrast image of a document page, likely a technical drawing or data table. The images are very small and difficult to read, but they appear to be organized in a structured manner. The right side of the card is mostly blank, with some faint markings and a small grid of frames in the bottom right corner.

B01

EOF1DCKBFCSEQ

00010000

770323

PDP10 411

02HDR1DCMFA0SEQ

00010000

770323

.REM *

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DCMFA-D-D

PRODUCT NAME: COMBINED MS-11 (MOS PARITY) AND MF11-LP, MA11-P (CORE)
PARITY MEMORY TESTS

DATE RELEASED: MARCH, 1977

MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1973, 1977
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

1.0 ABSTRACT

THIS PROGRAM LOCATES THE PARITY MEMORY REGISTERS FOR BOTH THE CORE AND MOS PARITY MEMORIES AND PERFORMS A CHECK OF THE BITS IN EACH. IT THEN CREATES A MAP SHOWING THE MEMORY CONTROLLED BY EACH PARITY REGISTER. THE PARITY REGISTERS AND THE MEMORY ARE THEN TESTED USING THE INFORMATION IN THE MAP.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 WITH MF11-LP OR MA11-P PARITY MEMORY (CORE), MS-11 (MOS) PARITY MEMORY

2.2 STORAGE

THE PROGRAM REQUIRES 4K OF MEMORY TO LOAD AND 8K TO RUN.

3.0 LOADING PROCEDURE

LOAD PROGRAM INTO MEMORY USING ABS LOADER.

4.0 STARTING PROCEDURE

4.1 STARTING ADDRESSES

200= NORMAL (WORST CASE) TESTING
210= ROUTINE TO RESTORE THE LOADER
220= ROUTINE TO SCAN FOR BAD PARITY
230= RESTART OF NORMAL TESTING- USES PREVIOUS MAP OF PARITY MEMORY

4.2.1 PROGRAM AND/OR OPERATOR ACTION

LOAD STARTING ADDRESS.
SET DESIRED SWITCH REGISTER SETTINGS (SEE 5.1- ALL DOWN FOR WORST CASE).
PRESS START.
IF SA 200 OR RESTART ADDRESS 230 IS USED, THE BELL WILL RING AT THE COMPLETION OF EACH PASS AND END PASS= XXX WILL BE TYPED (WHERE XXX IS THE NUMBER OF PASSES COMPLETED SINCE THE PROGRAM WAS LAST STARTED).
IF SA 210 OR SA 220 IS USED, THE PROGRAM WILL HALT WHEN DONE.

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

THE DIAGNOSTIC IS DESIGNED TO USE HARDWARE SWR FOR SYSTEMS HAVING THIS REGISTER, HOWEVER FOR SYSTEMS NOT HAVING HARDWARE SWITCH REGISTER IT WILL USE LOCATION 176 TO GIVE THE FOLLOWING OPTIONS:

SW 15=1 OR UP -- HALT ON ERROR
SW 14=1 OR UP -- SCOPE LOOP
SW 13=1 OR UP -- INHIBIT PRINTOUT
SW 11=1 OR UP -- INHIBIT ITERATIONS
SW 10=1 OR UP -- HALT AFTER LOCATING BAD PARITY BEFORE CORRECTING IT
(USED IN PARITY SCAN ROUTINE ONLY)

SW 09=1 OR UP -- HALT AFTER THE PARITY MEMORY MAP HAS BEEN PRINTED
(ALLOWS MANUAL CHANGES TO FORCE TESTING OF MEMORY
THAT WAS NOT LOCATED)
SW 08=1 OR UP -- HALT AT END OF PASS (IF HALTED ELSEWHERE, THE
PROGRAM MAY BE RELOCATED TO BANK 1, BAD PARITY MAY
EXIST IN MEMORY, AND/OR WRITE WRONG PARITY MAY
BE SET)

5.2 SUBROUTINE ABSTRACTS

5.2.1 BEGIN SA 200, RESTART 230

5.2.2 SCOPE

THIS SUBROUTINE CALL IS PLACED BETWEEN EACH SUBTEST IN THE INSTRUCTION SECTION. IT RECORDS THE STARTING ADDRESS OF EACH SUBTEST AS IT IS BEING ENTERED. IF A SCOPE LOOP IS REQUESTED, IT WILL JUMP TO THE START OF THE SUBTEST THAT THE SCOPE LOOP IS REQUESTED FOR. IF SCOPE LOOP IS NOT REQUESTED, THERE WILL BE 64 ITERATIONS OF THAT SUBTEST BEFORE THE NEXT SUBTEST IS ENTERED (EXCEPT IN THOSE ROUTINES WHERE IMAX IS CHANGED). SWITCH 11 ON A ONE INHIBITS ITERATION OF SUBTESTS.

5.2.3 ERROR HANDLERS (ERRST,ERRP,ERR)

THESE ROUTINES ARE CALLED VIA EMTS TO PRINT OUT ERROR INFORMATION.
(SEE 6.0 FOR DESCRIPTION OF ERROR INFORMATION)

5.2.4 PSCAN (SCAN MEMORY FOR BAD PARITY)

THIS ROUTINE READS ALL LOCATIONS IN MEMORY AND PRINTS OUT THE PHYSICAL ADDRESSES (18 BITS) OF THOSE LOCATIONS CONTAINING BAD PARITY. IT IS UTILIZED WITHIN THE PROGRAM WHILE EXERCISING MEMORY IF A PARITY ERROR OCCURS UNEXPECTEDLY, AND MAY ALSO BE CALLED USING STARTING ADDRESS 220.

5.2.5 \$TYPE (ASCII MESSAGE TYPEOUT ROUTINE)

THIS IS THE STANDARD TYPEOUT ROUTINE, ALLOWING PATCHING TO UTILIZE OUTPUT DEVICES OTHER THAN THE ASR 33. \$NULL CONTAINS THE VALUE TO BE USED AS A FILLER CHARACTER, AND \$FILLS CONTAINS A NUMBER INDICATING THE NUMBER OF FILLER CHARACTERS REQUIRED. TPS AND TPB CONTAIN THE STATUS AND BUFFER REGISTER ADDRESSES OF THE OUTPUT DEVICE.

5.2.6 TRAPCATCHER

THIS IS A SERIES OF INSTRUCTIONS STARTING AT LOCATION 0 DESIGNED TO DETECT AND ISOLATE UNEXPECTED TRAPS AND INTERRUPTS TO THE TRAP AND INTERRUPT VECTOR AREA OF MEMORY.

EACH VECTOR ENTRANCE ADDRESS IS LOADED WITH THE ADDRESS OF THE NEXT LOCATION. THE NEXT LOCATION IS LOADED WITH A HALT (000000). THUS AN ILLEGAL TRAP OR INTERRUPT WILL CAUSE A HALT AT THE TRAP LOCATION PLUS TWO.

IF A HALT OCCURS IN THE TRAP OR INTERRUPT AREA, EXAMINE REGISTER SIX.

IT WILL CONTAIN THE CURRENT STACK ADDRESS. THE CONTENTS OF THE CURRENT STACK ADDRESS IS THE VALUE OF THE LOCATION COUNTER WHEN THE TRAP OR INTERRUPT OCCURRED.

5.3 PROGRAM AND/OR OPERATOR ACTION

5.3.1 ALTERING THE PARITY MEMORY MAP

IF THE MAP TYPED AT RUN TIME DOES NOT AGREE WITH THE HARDWARE PRESENT THE MAP CAN MANUALLY BE CHANGED TO ALLOW TESTING OF PARITY MEMORY THAT THE MAPPER DID NOT FIND. SETTING SWITCH 9 TO A 1 WILL CAUSE THE PROGRAM TO HALT AFTER THE MAP IS TYPED. AFTER THE HALT, MODIFY THE MAP AS DESIRED (SEE THE DESCRIPTION IN THE LISTING- THE MAP BEGINS AT LOCATION 600). THEN PRESS CONTINUE. THE NEW MAP WILL BE PRINTED, AND IF SW9 IS STILL SET THE PROCESS WILL BE REPEATED. IF SW9 IS NOT SET, THE PROGRAM WILL TEST PARITY MEMORY USING THE NEW MAP.

5.3.2 STOPPING THE PROGRAM

BECAUSE THE PROGRAM RELOCATES ITSELF TO BANK 1 WHILE TESTING BANK 0, A SWITCH IS PROVIDED TO HALT THE PROGRAM AT THE END OF A PASS. SETTING THIS SWITCH (SW8) WILL CAUSE THE PROGRAM TO HALT IN BANK 0 AT THE END OF THE CURRENT PASS (AFTER OUTPUTTING THE END OF PASS MESSAGE).

6.0 ERRORS

6.1 ERROR PRINTOUTS

THERE ARE THREE TYPES OF ERROR MESSAGES USING COMBINATIONS OF THE FOLLOWING ERROR TYPE ROUTINES.

PC=ZZZZZZ PC OF FAILING ERROR CALL. REFER TO THIS ADDRESS IN THE LISTING FOR AN EXPLANATION OF THE ERROR.

ICNT=YYYYYY CURRENT ITERATION COUNT OF FAILING TEST.
MPR=XXXXXX ADDRESS OF PARITY REGISTER UNDER TEST.
MPR DATA=VVVVVV CONTENTS OF PARITY REGISTER UNDER TEST.
TEST LOC=XXXXXX MEMORY LOCATION UNDER TEST
S/B: XXXXXX CONTENTS OF MEMORY LOCATION SHOULD BE.
WAS: XXXXXX CONTENTS OF MEMORY LOCATION WAS.

6.2 DETERMINING ADDRESS OF TEST LOCATION WHEN KT11 IS PRESENT

IN MOST OF THE SUBTESTS, IF A KT11 IS PRESENT IT IS USED. IN ALL CASES IN THIS PROGRAM, WHEN THE KT11 IS ON, KERNEL PAGE 0 IS USED TO REFERENCE BANK 0 AND KERNEL PAGE 7 IS USED TO REFERENCE THE EXTERNAL BANK. IN MOST CASES, KERNEL PAGE 1 IS USED TO REFERENCE THE MEMORY CURRENTLY UNDER TEST. SINCE THE USE OF THE MEMORY MANAGEMENT OPTION IS SIMILAR THROUGHOUT THE PROGRAM, IT IS EASY TO DETERMINE THE ACTUAL (PHYSICAL) MEMORY ADDRESS BEING TESTED.

TO CALCULATE A PHYSICAL ADDRESS, ADD THE STARTING ADDRESS OF THE BANK BEING TESTED TO THE OFFSET WHICH GIVES THE ADDRESS WITHIN THE

BANK. SINCE IN THIS PROGRAM ALL RELOCATED MEMORY TESTING IS DONE THRU KERNEL PAGE 1, KERNEL PAGE ADDRESS REGISTER 1 (ADDRESS 772342) WILL ALWAYS CONTAIN THE STARTING ADDRESS OF THE BANK. ACTUALLY, KERNEL PAGE ADDRESS REGISTER 1 (KPAR1) CONTAINS JUST THE TOP 12 BITS OF THE BANK STARTING ADDRESS. ADDING TWO ZEROES (OCTAL) TO THE RIGHT OF THIS VALUE WILL GIVE YOU THE FULL 18 BIT ADDRESS OF THE BANK. THE VIRTUAL ADDRESS USED TO REFERENCE THIS BANK UNDER TEST WILL ALWAYS START WITH 001 (BINARY, TOP 3 OF 16 BITS). THIS REFERENCES PAGE 1. THE LOWER 13 BITS GIVE THE ADDRESS WITHIN THE BANK- ADD THEM TO THE STARTING ADDRESS OF THE BANK TO GET THE FULL 18 BIT PHYSICAL ADDRESS.

FOR EXAMPLE, AN ERROR COMMENT MAY SAY "R1 CONTAINS THE ADDRESS OF THE TEST LOCATION (VIRTUAL THRU KERNEL PAGE 1 IF KTI1 PRESENT)." R1 MIGHT CONTAIN 32000, AND KERNEL PAGE ADDRESS REGISTER 1 (LOCATION 772342) MIGHT CONTAIN 2400. FIRST GET THE STARTING ADDRESS OF THE BANK BY ADDING 2 ZEROES TO THE RIGHT OF THE NUMBER IN KPAR1. THUS THE VALUE 2400 INDICATES THAT THE BANK STARTS AT 240000. SECOND, CALCULATE THE OFFSET WITHIN THE BANK. THE VIRTUAL ADDRESS 32000 BREAKS DOWN INTO 1(TOP 3 BITS) WHICH REFERENCES KPAR1, AND 12000 (LOWER 13 BITS) WHICH IS THE OFFSET. ADD THE OFFSET (12000) TO THE BANK ADDRESS (240000) TO GET THE ACTUAL PHYSICAL ADDRESS BEING TESTED (252000).

6.3 ERROR RECOVERY

IN GENERAL, TEST FAILURES WILL PRINTOUT AN ERROR MESSAGE AND CONTINUE. IF THE HALT ON ERROR SWITCH IS SET, HITTING CONTINUE WILL RECOVER. IF THE PROGRAM HANGS UP IN A LOOP, THE ERROR IS LIKELY TO BE A SIGNAL WHICH WAS NEVER RECEIVED. IF A HALT OCCURS IN THE TRAP AND VECTOR AREA THE PROGRAM MUST BE RESTARTED. IF THE PROGRAM HALTS IN THE MAIN FLOW, CONSULT THE LISTING IF NO MESSAGE IS TYPED OUT.

6.4 ERRORS WHILE TESTING BANK ZERO (ERROR PC VALUES ABOVE 20000)

TEST20 AND TEST21 CHECK BANK 0 IF IT HAS PARITY MEMORY. TO DO THIS, THE CODE IS RELOCATED TO AND EXECUTED FROM BANK 1. THE ERROR PRINTOUTS WILL THUS GIVE THE PC IN BANK 1 OF THE ERROR CALL. SINCE ALL LOCATIONS HAVE BEEN MOVED UP 20000, SUBTRACT 20000 FROM THE ERROR PC TO GET THE ADDRESS IN THE LISTING WHICH CORRESPONDS TO THE PRINTOUT.

7.0 RESTRICTIONS

THE PROGRAM REQUIRES A MINIMUM OF 8K MEMORY TO RUN. WHEN RUNNING UNDER ACT BANK 0 WILL NOT BE TESTED.

7.1 STARTING PROCEDURE

PROGRAM MUST BE LOADED INTO LOWER 4K OF MEMORY.

7.2 OPERATING RESTRICTION- AVOID USING THE "HALT" SWITCH

IF THE PROGRAM IS HALTED AT A RANDOM POINT DURING EXECUTION, SEVERAL PROBLEMS MAY ARISE. THE PROGRAM MAY BE RELOCATED TO BANK 1 AT THE

TIME IT IS STOPPED, IN WHICH CASE NONE OF THE STANDARD STARTING ADDRESSES WILL WORK. WRITE WRONG PARITY MAY BE SET, IN WHICH CASE YOU MAY ENTER BAD PARITY WHILE PATCHING. AND MEMORY MAY CONTAIN BAD PARITY SINCE YOU MAY BE IN THE MIDDLE OF A TEST WHICH UTILIZES WRITE WRONG PARITY. IT IS THEREFORE STRONGLY RECOMMENDED THAT YOU HALT THE PROGRAM VIA THE "HALT AT END OF PASS" SWITCH (SW8) OR THE "HALT ON ERROR" SWITCH (SW15) RATHER THAN VIA THE HALT/ENABLE SWITCH.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

EXECUTION TIME DEPENDS ON THE AMOUNT OF PARITY MEMORY UNDER TEST. IT TAKES ABOUT 1 MINUTE TO TEST 24K OF PARITY MEMORY (1 PASS).

8.2 STACK POINTERS

THE KERNEL STACK POINTER IS INITIALIZED TO 510.

9.0 PROGRAM DESCRIPTION

THIS PROGRAM FIRST LOCATES MA11 & MF11 CORE PARITY AND MS-11 MOS PARITY CONTROL REGISTERS BY ADDRESSING EACH POSSIBLE REGISTER ADDRESS AND CHECKING THOSE WHICH DO NOT TIME OUT. ON DETECTING THE PRESENCE OF A PARITY REGISTER THE PROGRAM CHECKS IF IT IS A CORE PARITY OR A MOS PARITY REGISTER AND ACCORDINGLY STORES THIS INFORMATION IN AN INDICATOR (INDCO-INDC15) THE ADDRESSES OF THE REGISTERS ARE RECORDED AND OUTPUT TO THE CONSOLE DEVICE, AND THEN THE REGISTERS ARE CHECKED TO SEE THAT THE CORRECT BITS ARE R/W. RESET IS USED TO TEST THE EFFECT OF INIT. PARITY MEMORY IS THEN LOCATED BY SETTING WRITE WRONG PARITY IN ALL REGISTERS AND WRITING AND READING THE FIRST 4 ADDRESSES IN EACH 4K. EACH TIME A PARITY REGISTER RECORDS A PARITY ERROR, THE MAP IS ALTERED TO INDICATE THAT THAT REGISTER CONTROLS THE MEMORY BEING ADDRESSED. THE FINAL MAP IS PRINTED AND THEN THE PARITY CONTROL LOGIC IS CHECKED USING THE PARITY MEMORY FOUND. SEVERAL PATTERNS ARE WRITTEN INTO EACH PARITY MEMORY LOCATION TO SEE THAT NO PARITY ERRORS ARE CREATED. FINALLY, EACH BYTE OF PARITY MEMORY IS WRITTEN WITH BOTH GOOD AND BAD PARITY TO SHOW THAT THE PARITY BITS CAN BE TOGGLED AND SENSED. SINCE THIS IS A COMBINED DIAGNOSTIC, AS FAR AS POSSIBLE COMMON TESTS ARE USED FOR BOTH CORE AND MOS. ONLY WHERE THE MOS CONTROLLER DEFERS FUNCTIONALLY FROM THE CORE, THE INDICATOR IS CHECKED FOR MOS OR CORE AND THE MEMORY IN QUESTION IS TESTED ACCORDINGLY. A DETAILED EXPLANATION OF THE MAP IS GIVEN IN THE LISTING (PAGE 9-12). THE DISPLAY REGISTER CONTAINS THE NUMBER OF THE TEST BEING EXECUTED.

*

:MEMORY PARITY TEST
:MAINDEC-11-DCMFA-D
:COPYRIGHT 1973, 1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
:AUTHOR: JIM KAPADIA

327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378

:SWITCH REGISTER SWITCH OPTIONS (SWITCH SET TO A 1)
:SR15 - HALT ON ERROR
:SR14 - SCOPE
:SR13 - INHIBIT PRINTOUT
:SR11 - INHIBIT ITERATIONS
:SR10 - HALT AFTER LOCATING BAD PARITY BEFORE CORRECTING IT
:SR09 - HALT AFTER TYPING PARITY MEMORY MAP (ALLOWS MANUAL
: CHANGES TO BE MADE TO THE MAP TO FORCE TESTING OF
: MEMORY THAT WAS NOT LOCATED)
:SR08 - HALT AT END OF PASS (IF HALTED ELSEWHERE, THE PROGRAM
: MAY BE RELOCATED TO BANK1, WRITE WRONG PARITY MAY BE SET,
: AND/OR BAD PARITY MAY EXIST IN THE PARITY MEMORY).

:SYMBOL DEFINITIONS

BIT0=1
BIT1=2
BIT2=4
BIT3=10
BIT4=20
BIT5=40
BIT6=100
BIT7=200
BIT8=400
BIT9=1000
BIT10=2000
BIT11=4000
BIT12=10000
BIT13=20000
BIT14=40000
BIT15=100000
AE=1
WMP=4
ADRS=77400
PERR=100000
DSWR=177570
DDISP=177570
PS=177776
PC=%7
SP=%6
NOP=240
OPEN=0
STKPT=TSTX
RO=%0
R1=%1
R2=%2

:BIT DEFINITIONS

:ACTION ENABLE
:WRITE WRONG PARITY
:ADDRESS OF ERROR
:PARITY ERROR BIT
:HARDWARE SWITCH REGISTER
:HARDWARE DISPLAY REGISTER

000001
000002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000
000001
000004
077400
100000
177570
177570
177776
000007
000006
000240
000000
000510
000000
000001
000002

379 000003
380 000004
381 000005
382 000006
383 000114
384 177572

R3=%3
R4=%4
R5=%5
R6=%6

PARVEC=114
SRO=177572

;PARITY ERROR TRAP VECTOR
;ADDRESS OF MEM MGMT REGISTER SRO

;MACRO DEFINITIONS

;TRAPCATCHER (.+2,HALT) LOADED INTO LOCATIONS 000-576

;LOAD EMT VECTOR

398 000030
399 000030 015550
400 000032 000340
401 000046
402 000046 011444
403 000052
404 000052 040000

. =30
EMTINT
340
. =46
\$ENDAD
. =52
BIT14

;LOAD STARTING ADDRESS AREA

407 000174
408 000174 000000
409 000176 000000
410 000200 000167 001226
411 000210
412 000210 000167 013722
413 000220
414 000220 000167 001126
415 000230
416 000230 000167 001036
417
418 000510
419

. =174
DISPREG: .WORD 0
SWREG: .WORD 0
JMP START
. =210
JMP RSTLDR
. =220
JMP SCAN
. =230
JMP RSTART
. =510

;THIS IS THE SOFTWARE DISPLAY REGISTER
;THIS IS THE SOFTWARE SWITCH REGISTER
;GO TO START OF PROGRAM
;GO RESTORE THE LOADERS
;SCAN FOR BAD PARITY
;RESTART WITHOUT RETYPING MAP INFORMATION

;GENERAL DATA AREA

424 000510 000000
425 000512 000000
426 000514 000000
427 000516 000000
428 000520 000000
429 000522 000000
430 000524 000000
431 000526 000000
432 000530 000000
433 000532 000000
434

†STX: 0
FTITLE: 0
TEMPX: 0
ADRPT: 0
BITPT: 0
TRFLG: 0
TYFLG: 0
TYCOR: 0
HIADR: 0
TSTLOC: 0

;TITLE PRINTED = 1
;MAPPING- ADDRESS POINTER
;MAPPING- BIT POINTER INDICATING BANK
;MAPPING- TRANSITION FLAG
;MAPPING- TYPED FLAG
;MAPPING- K CORE ACCUMULATOR
;USED TO CHECK WHEN DONE TESTING A BANK
;LOADED WITH ADDRESS OF LOCATION UNDER
;TEST IN SOME SUBTESTS

```

435 000534 000000      SHDBE: 0          ;VALUE EXPECTED
436 000536 000000      WAS: 0           ;ACTUAL VALUE FOUND
437 000540 000000      TRDATA: 0
438 000542 000000      MPROK: 0
439 000544 000000      PASCNT: 0       ;PASS COUNT
440 000546 000000      TBANK: 0
441 000550 000000      MEMUT: 0
442 000552 000000      NOKT: 0        ;SET TO INDICATE NO KT11 PRESENT
443 000554 000000      HIWORD: 0
444 000556 000000      LOWFLG: 0
445 000560 000000      ODDFLG: 0     ;IF SET INDICATES TESTING HIGH BYTE
446                                     ;OF MEMORY LOCATION
447 000562 000000      TEMP: 0
448 000564 000000      MTYFG: 0     ;SET TO INDICATE MAP OF PARITY MEMORY
449                                     ;ALREADY TYPED
450 000566 000000      RELOC: 0
451
452
453
454 ;MEMORY PARITY CONTROL REGISTER ADDRESSES
455 ;THE LEAST SIGNIFICANT BIT IN THE DEVICE ADDRESS IS SET TO A ONE(1)
456 ;IF THE CONTROL IS FOUND NOT TO BE PRESENT. THE MEMORY PRESENT UNDER
457 ;CONTROL OF EACH CONTROLLER IS REPRESENTED BY 2 OCTAL WORDS. EACH BIT
458 ;REPRESENTS A 4K BLOCK, I.E. BIT0= 0-4K, BIT1= 4-8K, BIT15= 60-64K.
459 ;THE LOW BYTE OF THE LAST WORD FOR EACH REGISTER INDICATES THE OFFSET (0,2,4 OR 6)
460 ;FOR THE FIRST ADDRESS THAT ACTUALLY CORRESPONDED TO THE REGISTER. THE HIGH BYTE GETS
461 ;SET TO 1 TO INDICATE THAT A MEMORY ADDRESS HAS BEEN FOUND FOR THAT REGISTER.
462 ;FOR EXAMPLE, SAY THAT MPRO AND MPRI EXIST, CONTROLLING INTERLEAVED MEMORY
463 ;FROM 0 TO 16K, AND THAT MPRO CONTROLS THE ADDRESSES ENDING IN 0 AND 4.
464 ;THE MAP WOULD THEN LOOK AS FOLLOWS:
465 ;       MPRO: 172100          ;BIT 0 IS CLEAR SINCE REGISTER IS PRESENT
466 ;               17          ;REGISTER CONTROLS 1ST 16K (=4 BANKS)
467 ;               0
468 ;               400          ;LOW BYTE SHOWS THAT FIRST ADDRESS
469 ;                               ;ENDS IN 0 (OCTAL)
470 ;                               ;HIGH BYTE CONTAINS A 1 TO INDICATE
471 ;                               ;THAT AN ADDRESS WAS FOUND
472 ;       MPRI: 172102          ;BIT 0 IS CLEAR SINCE REGISTER IS PRESENT
473 ;               17          ;REGISTER CONTROLS 1ST 16K
474 ;               0
475 ;               402          ;LOW BYTE INDICATES THAT THE FIRST
476 ;                               ;MEMORY ADDRESS ENDS IN 2 (OCTAL)
477 ;                               ;HIGH BYTE CONTAINS A 1 TO INDICATE
478 ;                               ;THAT AN ADDRESS WAS FOUND
479 ;THE REST OF THE MAP WOULD APPEAR AS IN THE LISTING
480
481 000570 172101      MPRO: 172100+1   ;PARITY STATUS REGISTERS
482 000572 000000      0          ;0-64K PARITY MEM UNDER THIS CONTROL
483 000574 000000      0          ;64-124K PARITY MEM UNDER THIS CONTROL
484 000576 000000      0          ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
485 000600 172103      MPRI: 172102+1
486 000602 000000      0          ;0-64K PARITY MEM UNDER THIS CONTROL
487 000604 000000      0          ;64-124K PARITY MEM UNDER THIS CONTROL
488 000606 000000      0          ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
489 000610 172105      MPR2: 172104+1
490 000612 000000      0          ;0-64K PARITY MEM UNDER THIS CONTROL

```

491	000614	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
492	000616	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
493	000620	172107	MPR3:	172106+1	
494	000622	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
495	000624	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
496	000626	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
497	000630	172111	MPR4:	172110+1	
498	000632	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
499	000634	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
500	000636	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
501	000640	172113	MPR5:	172112+1	
502	000642	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
503	000644	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
504	000646	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
505	000650	172115	MPR6:	172114+1	
506	000652	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
507	000654	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
508	000656	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
509	000660	172117	MPR7:	172116+1	
510	000662	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
511	000664	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
512	000666	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
513	000670	172121	MPR8:	172120+1	
514	000672	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
515	000674	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
516	000676	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
517	000700	172123	MPR9:	172122+1	
518	000702	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
519	000704	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
520	000706	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
521	000710	172125	MPR10:	172124+1	
522	000712	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
523	000714	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
524	000716	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
525	000720	172127	MPR11:	172126+1	
526	000722	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
527	000724	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
528	000726	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
529	000730	172131	MPR12:	172130+1	
530	000732	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
531	000734	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
532	000736	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
533	000740	172133	MPR13:	172132+1	
534	000742	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
535	000744	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
536	000746	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
537	000750	172135	MPR14:	172134+1	
538	000752	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
539	000754	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
540	000756	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
541	000760	172137	MPR15:	172136+1	
542	000762	000000		0	;0-64K PARITY MEM UNDER THIS CONTROL
543	000764	000000		0	;64-124K PARITY MEM UNDER THIS CONTROL
544	000766	000000		0	;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
545					
546	000770	000000	TREG:	0	;PARITY REGISTER UNDER TEST

MO1

DCMFA-D MEMORY PARITY TEST
DCMFAD.P11 25-OCT-76 22:48

MACY11 27(1006) 25-OCT-76 22:52 PAGE 11

547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602

; INDICATORS FOR CORE OR MOS PARITY REGISTER:
; EACH INDICATOR REFERS TO A PARTICULAR PARITY REGISTER. IF IT IS
; A CORE PARITY REGISTER THEN A '1' IS STORED IN THE INDICATOR.
; IF IT IS A MOS PARITY REGISTER THEN '-1' GETS STORED.
; EX= IF MPRO (172100) IS FOR CORE AND MPR1 (172102) IS FOR MOS
; THEN THE INDICATOR MAP WILL LOOK AS FOLLOWING:

; INDC0: 000001
; INDC1: 177777

INDC0: 0	; CORE-MOS PARITY INDICATOR FOR MPR
INDC1: 0	; CORE-MOS PARITY INDICATOR FOR MPR1
INDC2: 0	; CORE-MOS PARITY INDICATOR FOR MPR2
INDC3: 0	; CORE-MOS PARITY INDICATOR FOR MPR3
INDC4: 0	; FOR MPR4
INDC5: 0	; FOR MPR5
INDC6: 0	; FOR MPR6
INDC7: 0	; FOR MPR7
INDC8: 0	; FOR MPR8
INDC9: 0	; FOR MPR9
INDC10: 0	; FOR MPR10
INDC11: 0	; FOR MPR11
INDC12: 0	; FOR MPR12
INDC13: 0	; FOR MPR13
INDC14: 0	; FOR MPR14
INDC15: 0	; FOR MPR15
RESRVD: 0	

; BIT POSITIONS WHICH ARE RESERVED
; FOR FUTURE USE IN PARITY REGISTERS
; CORE PARITY
; MOS PARITY

RESVC: 70032
RESVM: 77772

; PARITY PATTERNS

PARPAT: 125325
152652
052452
025125
102070
072527
177777
107030
152525
0
0

; EVEN, ODD BYTES
; ODD, EVEN
; EVEN, ODD
; ODD, EVEN
; EVEN, EVEN
; ODD, ODD
; EVEN, EVEN
; ODD, ODD
; ODD, EVEN
; EXTRA PATTERN AREA
; TERMINATOR, DO NOT USE THIS LOC

; THIS IS A MAP OF THE TOTAL MEMORY PRESENT IN THE SYSTEM.

MEML: 0 ; 0-64K MEM PRESENT IN 4K CONTIGUOUS BLOCKS
MEMH: 0 ; 64-124 MEM PRESENT IN 4K CONTIGUOUS BLOCKS

; THIS IS A MAP OF THE TOTAL PARITY MEMORY PRESENT IN THE SYSTEM.

659
660 001222 000000
661 001224 000000
662 001226 177600
663 001230 172200
664 001232 172300
665 001234 172300
666 001236 172302
667 001240 172304
668 001242 172316
669 001244 172340
670 001246 172342
671 001250 172344
672 001252 172356
673 001254 000000
674 001256 000000
675 001260 000000
676 001262 000000
677 001264 000000
678 001266 000000
679 001270 000000
680
681
682
683
684 001272 012706 000510
685 001276 012767 000001 177260
686 001304 005067 177234
687 001310 012737 015450 000024
688 001316 012737 000340 000026
689 001324 005067 177160
690 001330 005037 177776
691 001334 012737 000006 000004
692 001342 005037 000006
693 001346 000167 000500
694
695
696
697
698
699
700 001352 012706 000510
701 001356 005767 177130
702
703
704 001362 001006
705 001364 005267 177630
706 001370 000167 000202
707 001374 005067 177620
708 001400 004767 010110
709 001404 004767 012622
710 001410 104000
711 001412 017230
712 001414 005767 177132
713 001420 001002
714 001422 005037 177572

KTSTART: 0
ADRTYP: 0
PDRTAB: 177600
172200
PDREND: 172300
KPDRO: 172300
KPDR1: 172302
KPDR2: 172304
KPDR7: 172316
KPAR0: 172340
KPAR1: 172342
KPAR2: 172344
KPAR7: 172356
SPSAV: 0
ROSAV: 0
RISAV: 0
R2SAV: 0
R3SAV: 0
R4SAV: 0
R5SAV: 0

;SCAN ROUTINE (SA=220)

;KERNEL PAGE DESCRIPTOR REGISTER ADDRESSES

;KERNEL PAGE ADDRESS REGISTER ADDRESSES

;REGISTER SAVE LOCATIONS

;ROUTINE TO RESTART WITHOUT RETYPING MAP AFTER TEST HAS BEEN RUNNING
RSTART: MOV #STKPT,SP ;SET UP STACK POINTER
MOV #1,MTYFG ;SET FLAG TO INDICATE MAP HAS BEEN TYPED
CLR PASCNT ;INITIALIZE PASS COUNT
MOV #PWRDN, @#24
MOV #340, @#26
CLR TSTX
CLR @#PS ;CLEAR PROCESSOR STATUS REGISTER
MOV #6, @#4
CLR @#6
JMP BEGIN

;ROUTINE TO SCAN ALL MEMORY FOR BAD PARITY AND TYPE 18 BIT ADDRESSES OF BAD
;LOCATIONS
SCAN: MOV #STKPT,SP ;SETUP STACK POINTER
TST FTITLE ;IF TITLE HAS BEEN PRINTED, REGISTERS
;HAVE ALREADY BEEN LOCATED- GO
;AND LOCATE IF NOT ALREADY DONE
;BRANCH, REGISTERS HAVE ALREADY BEEN LOCATED
;INCREMENT SCNFLG
;GO TO LOCATE THE REGISTERS
;RETURN HERE AFTER LOCATING THE REGISTERS
SCANA: CLR SCNFLG ;SETUP MEMORY MAP
SCANB: JSR PC,MAPMEM ;SCAN FOR BAD PARITY
;TYPE MESSAGE "BAD PARITY SCAN COMPLETE"
TYPE
PSMSG
TST NOKT
BNE .+6
CLR @#SRO ;TURN OFF KT11 IF PRESENT

```

715 001426 000000          HALT          ;END OF PARITY SCAN
716 001430 000750          BR           SCAN
717
718
719
720
721          :NORMAL STARTUP
722 001432 012706 000510  START:  MOV      #STKPT, SP          ;SET UP STACK POINTER
723 001436 005067 177122          CLR      MTYFG          ;CLEAR FLAG WHICH INDICATES MAP TYPED
724 001442 005067 177076          CLR      PASCNT        ;INITIALIZE PASS COUNT
725 001446 012737 015450 000024  MOV      #PWRDN, @#24   ;SETUP POWER FAIL RETURN
726 001454 012737 000340 000026  MOV      #340, @#26
727 001462 013746 000004          MOV      @#4, -(SP)    ;SAVE THE ERROR VECTOR
728 001466 013746 000006          MOV      @#6, -(SP)
729 001472 012737 001506 000004  MOV      #25, @#4      ;SET UP TIME OUT VECTOR
730 001500 005777 177374          TST     @SWR          ;TRY TO REFERENCE THE SWITCH REGISTER
731 001504 000407          BR           45        ;BRANCH IF NO TIME OUT
732 001506 012767 000176 177364 25:  MOV      #SWREG, SWR   ;POINT TO THE SOFTWARE SWITCH REGISTER
733 001514 012767 000174 177360  MOV      #DISPREG, DISPLAY ;POINT TO THE SOFTWARE DISPLAY REG.
734 001522 022626          CMP      (SP)+, (SP)+ ;RESTORE THE STACK POINTER
735 001524 012637 000006 45:  MOV      (SP)+, @#6    ;RESTORE ERROR VECTOR
736 001530 012637 000004          MOV      (SP)+, @#4
737 001534 005067 176750          CLR      TSTX
738 001540 005767 176746 15:  TST     FTITLE        ;IS TITLE PRINTED YET?
739 001544 001014          BNE     START1       ;YES, SKIP OVER
740 001546 004767 012246          JSR     PC, SAVLDR   ;COPY LOADER TO LOWER 4K
741 001552 005267 176734          INC     FTITLE
742 001556 023737 000042 000046  CMP      @#42, @#46  ;SET FLAG
743 001564 001404          BEQ     START1       ;ARE WE IN ACT11 AUTOMATIC MODE?
744 001566 104000          TYPE
745 001570 016774          MTIT
746 001572 104000          TYPE
747 001574 017263          MLDRSV
748
749
750
751
752
753          ;SEARCH FOR PARITY REGISTERS PRESENT AND TYPE ADDRESSES OF THOSE FOUND
754          ;FAILURE TO LOCATE A REGISTER INDICATES THAT THE ADDRESS TIMED OUT OR THAT
755          ;BITS 5-7 IN THE REGISTER DID NOT SET
756 001576 104000  START1: TYPE          ;TYPE "MEMORY PARITY REGISTERS PRESENT ARE:"
757 001600 016544          MPRS
758 001602 005067 176734          CLR      MPROK        ;CLEAR MPR FLAG
759 001606 012702 000570          MOV      #MPRO, R2    ;SET UP POINTERS
760 001612 012703 000772          MOV      #INDC0, R3   ;POINTER TO CORE-MOS
761 001616 012737 001756 000004  MOV      #GMPRB, @#4  ;SET UP TIMEOUT TRAP RETURN
762 001624 005037 000006          CLR      @#6
763 001630 042712 000001  GMPRA: BIC      @1(2)    ;CLEAR FLAG BIT IN TABLE
764 001634 005062 000002          CLR      2(R2)       ;INITIALIZE LOCATIONS IN THE TABLE
765 001640 005062 000004          CLR      4(R2)
766 001644 005062 000006          CLR      6(R2)
767 001650 005772 000000          TST     @2(2)
768 001654 052772 000340 000000  BIS      #340, @2(2)  ;DOES THIS MPR EXIST? (IF NO, TIMES OUT)
769 001662 032772 000340 000000  BIT      #340, @2(2)  ;YES- IS IT AN MF11-LP OR MA11-P CORE PARITY REG
770 001670 001414          BEQ     15          ;NO, IS IT A MOS-11 PARITY REGISTER? BRANCH

```

```

771 001672 011267 176616      MOV      (2),TEMPX      ;YES- PRINT REGISTER ADDRESS
772 001676 004567 013232      JSR      RS,0ACNV      ;(GET ASCII)
773 001702 000514                TEMPX
774 001704 017355                MPRCOR
775 001706 000006                6
776 001710 104000                TYPE                ;(TYPE ADDRESS)
777 001712 017355                MPRCOR
778 001714 012713 000001      MOV      #1,(R3)        ;SET INDICATOR FOR CORE PARITY
779 001720 000413                BR
780 001722 011267 176566      15:     MOV      (2),TEMPX      ;IT IS A MOS REGISTER, PRINT ADDRESS
781 001726 004567 013202      JSR      RS,0ACNV      ;(GET ASCII)
782 001732 000514                TEMPX
783 001734 017416                MPRMOS
784 001736 000006                6
785 001740 104000                TYPE                ;(TYPE ADDRESS)
786 001742 017416                MPRMOS
787 001744 012713 177777      MOV      #-1,(R3)      ;SET INDICATOR FOR MOS PARITY
788 001750 005267 176566      25:     INC      MPROK      ;SET MPR REGISTER PRESENT FLAG
789 001754 000403                BR                ;SKIP NEXT
790 001756 022626      GMPRB:  CMP      (SP)+,(SP)+ ;RESTORE STACK POINTER
791 001760 052712 000001      BIS      #1,R2         ;SET FLAG INDICATING REGISTER NOT PRESENT
792 001764 062702 000010      GMPRC:  ADD      #10,R2  ;UPDATE POINTER
793 001770 005723                TST      (R3)+
794 001772 020227 000770      CMP      R2,#TREG     ;DONE YET?
795 001776 002714                BLT      GMPRA        ;NO, LOOP
796 002000 012737 000006 000004  MOV      #6,#4        ;YES, RESTORE TRAPCATCHER
797 002006 005767 177206      TST      SCNFLG      ;ARE YOU IN THE ROUTINE TO SCAN
798                                ;MEMORY FOR BAD PARITY-(SCAN)
799 002012 001402                BEQ      GMPRD        ;NO BRANCH TO CARRY ON NORMALLY
800 002014 000167 177354      JMP      SCANA        ;YES, GO BACK TO THE MEMORY
801                                ;SCAN ROUTINE
802 002020 005767 176516      GMPRD:  TST      MPROK      ;ANY PARITY REGISTERS PRESENT?
803 002024 001012                BNE      BEGIN        ;YES- GO TEST CONTROLS PRESENT
804 002026 104000                NOREG:  TYPE                ;NO- TYPE "NO PARITY REGISTER FOUND"
805 002030 016727                MTR
806 002032 005737 000042      TST      #4          ;LOADED BY MONITOR?
807 002036 001402                BEQ      .+6          ;NO, BRANCH
808 002040 000167 007370      JMP      LOGICAL      ;YES- EXIT, NO REGISTERS PRESENT
809 002044 000000                HALT
810 002046 000167 177360      JMP      START        ;NO REGISTERS TO TEST
811                                ;IF CONTINUED, TRY AGAIN
812 002052 012767 002076 014120  BEGIN:  MOV      #TEST1+2,RETURN ;SETUP SCOPE RETURN
813 002060 012767 000100 014106  MOV      #100,IMAX    ;MAXIMUM ITERATION COUNT
814 002066 012737 000006 000004  MOV      #6,#4        ;RESTORE TRAPCATCHER IN TIMEOUT VECTOR
815
816
817
818                                ;*****
819                                ;SHOW THAT BITS 0,2,5-11, AND 15 OF EACH CORE PARITY REGISTER PRESENT
820                                ;CAN BE SET AND CLEARED. BITS 0,2,15 OF EACH MOS PARITY REGISTER
821                                ;PRESENT CAN BE SET AND CLEARED
822                                ;*****
823 002074 104001                TEST1:  SCOPE
824 002076 012777 000001 176776  MOV      #1,#DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY
825 002104 012700 000570                MOV      #MPRO,R0     ;LOAD ADDRESS OF TABLE INTO R0
826 002110 012704 000772                MOV      #INDC0,R4    ;LOAD ADDRESS OF INDICATOR IN R4

```

```

827 002114 032710 000001      15:  BIT      #1, R0      ; IS THIS REGISTER PRESENT?
828 002120 001042              BNE      4$      ; NO- BRANCH TO GET NEXT ADDRESS
829 002122 011001              MOV      R0, R1  ; YES- LOAD R1 WITH ADDRESS OF
830                                ; PARITY REGISTER
831 002124 022714 000001              CMP      #1, (R4) ; IS THIS REGISTER CORE?
832 002130 001004              BNE      5$      ; NO
833 002132 016767 176676 176672      MOV      RESVC, RESRVD ; YES, CORE. STORE RESERVED BITS
834 002140 000403              BR       +10
835 002142 016767 176670 176662      5$:  MOV      RESVM, RESRVD ; MOS. STORE RESERVED BITS
836 002150 012702 000001              MOV      #1, R2  ; LOAD R2 WITH VALUE OF FIRST BIT
837                                ; TO BE TESTED
838 002154 005011              CLR      R1      ; INITIALIZE PARITY REGISTER
839 002156 011167 176606              MOV      R1, TREG ; READ CONTENTS OF PARITY REGISTER
840 002162 046767 176644 176600      BIC      RESRVD, TREG ; CLEAR BITS WHICH ARE RESERVED
841 002170 001401              BEQ     +4      ; CHECK OTHER BITS- BRANCH IF OK
842 002172 104002              ERROR     ; CLEAR INSTRUCTION DID NOT INITIALIZE
843                                ; ALL USED BITS IN PARITY REGISTER
844                                ; TO ZERO (R1 CONTAINS ADDRESS OF
845                                ; FAILING REGISTER)
846 002174 030267 176632      25:  BIT      R2, RESRVD ; IS THIS BIT RESERVED?
847 002200 001010              BNE      3$      ; YES- DON'T TEST IT SINCE IT
848                                ; MAY BE ZERO OR ONE
849 002202 010211              MOV      R2, R1  ; NO- SET THIS BIT IN THE PARITY REGISTER
850 002204 011103              MOV      R1, R3  ; READ AND SAVE CONTENTS OF PARITY REGISTER
851 002206 005011              CLR      R1      ; CLEAR PARITY REGISTER
852 002210 046703 176616      BIC      RESRVD, R3 ; CLEAR BIT LOCATIONS THAT ARE
853                                ; RESERVED
854 002214 020203              CMP      R2, R3  ; CHECK REST
855 002216 001401              BEQ     +4      ; BRANCH IF OK
856 002220 104002              ERROR     ; PARITY REGISTER WHOSE ADDRESS IS IN R1
857                                ; WAS INCORRECT AFTER THE VALUE IN R2
858                                ; WAS WRITTEN INTO IT. ACTUAL CONTENTS
859                                ; (WITH UNUSED BITS CLEARED) IS IN R3
860 002222 006302      3$:  ASL      R2      ; ROTATE BIT TO BE TESTED
861 002224 103363              BCC     2$      ; IF NOT DONE WITH ALL BIT POSITIONS
862                                ; GO TEST THIS ONE
863 002226 062700 000010      4$:  ADD      #10, R0 ; MOVE R0 TO POINT TO NEXT POSSIBLE ADDRESS
864 002232 005724              TST     (R4)+   ;
865                                ; OF A PARITY REGISTER
866 002234 020027 000770              CMP      R0, #TREG ; AT END OF TABLE?
867 002240 002725              BLT     1$      ; NO, BRANCH
868 002242 005067 176522              CLR      TREG
869
870 ; *****
871
872 ; SHOW THAT RESET CLEARS BITS 0,2, AND 15 OF EACH PARITY REGISTER
873 ; PRESENT.
874 ; *****
875 TEST2: SCOPE
876 002246 104001 000002 176624      MOV      #2, DISPLAY ; LOAD THE TEST NUMBER INTO THE DISPLAY
877 002250 012777 000002 176624      CLR      IMAX     ; DON'T ITERATE TEST
878 002256 005067 013712              MOV      #MPRO, R0 ; LOAD POINTER
879 002262 012700 000570              MOV      #INDCO, R3 ; POINTER TO INDICATOR
880 002266 012703 000772              MOV      #1, R0
881 002272 032710 000001      15:  BIT      #1, R0      ; IS THIS PARITY REGISTER PRESENT?
882 002276 001012              BNE     5$      ; NO BRANCH
883 002300 022713 000001              CMP      #1, (R3) ; IS THIS CORE OR MOS PARITY REGISTER?

```

883	002304	001404				BEQ	6\$;NO- BRANCH
884	002306	012770	100015	000000		MOV	#100015,2(R0)		;MOS-SET ALL DEFINED BITS TO 1
885	002314	000403				BR	.+10		
886	002316	012770	107745	000000	6\$:	MOV	#107745,2(R0)		;CORE- SET ALL DEFINED BITS TO 1
887	002324	062700	000010		5\$:	ADD	#10,R0		;MOVE POINTER TO POINT TO NEXT MPR ADDRESS
888	002330	005723				TST	(R3)+		;INCREMENT POINTER TO INDICATOR
889									;OF A PARITY REGISTER
890	002332	020027	000770			CMP	RO,#TREG		;AT END OF TABLE?
891									
892	002336	002755				BLT	1\$;NO- CONTINUE
893	002340	105767	176546			TSTB	\$TPFLG		;YES- TERMINAL AVAILABLE?
894	002344	001003				BNE	4\$;NO- BRANCH
895	002346	105777	176532			TSTB	\$TPS		;YES- WAIT FOR TERMINAL TO FINISH
896	002352	100375				BPL	.-4		
897	002354	000005			4\$:	RESET			;ISSUE INIT
898	002356	012700	000570			MOV	#MPRO,R0		;LOAD ADDRESS OF THE TABLE
899	002362	012703	000772			MOV	#INDCO,R3		;POINTER TO INDICATOR
900	002366	032710	000001		2\$:	BIT	#1,2RO		;IS THIS PARITY REGISTER PRESENT?
901	002372	001030				BNE	3\$;NO- BRANCH
902	002374	022713	000001			CMP	#1,(R3)		;IS THIS A CORE PAR REGISTER?
903	002400	001012				BNE	7\$;NO- BRANCH
904	002402	017002	000000			MOV	2(R0),R2		;YES, GET CONTENTS OF REGISTER
905	002406	005070	000000			CLR	2(R0)		;MAKE SURE THAT WWP AND AE ARE CLEAR
906	002412	042702	077772			BIC	#77772,R2		;MASK RESERVED BITS FOR CORE PAR
907									;ITY REGISTER. BITS 5-11(ADDRS
908									;BITS) ARE ALSO MASKED
909	002416	005702				TST	R2		;CHECK, IF REST WERE CLEARED
910	002420	001401				BEQ	.+4		
911	002422	104002				ERROR			;CORE PARITY REGISTER WHOSE ADDRESS IS
912									;POINTED TO BY RO WAS INCORRECT
913									;AFTER A RESET WAS ISSUED- CONTENTS
914									;AVED IN R2 WITH UNUSED BITS MASKED
915	002424	000411				BR	3\$-4		
916	002426	017002	000000		7\$:	MOV	2(R0),R2		;MOS, GET CONTENTS OF REGISTER
917	002432	005070	000000			CLR	2(R0)		;MAKE SURE THAT WWP BAE ARE CLEAR
918	002436	046702	176374			BIC	RESVM,R2		;MASK RESERVED BITS FOR MOS PAR REG
919	002442	005702				TST	R2		;CHECK REST
920	002444	001401				BEQ	.+4		;RESET DID CLEAR ALL BITS
921	002446	104002				ERROR			;MOS PARITY REGISTER WHOSE ADDRESS IS
922									;POINTED BY RO WAS INCORRECT. AFTER
923									;ISSUING RESET CONTENTS OF PAR REG
924									;WERE AS SHOWN IN R2(UNUSED BITS
925									;HAVE BEEN MASKED)
926	002450	005070	000000			CLR	2(R0)		;REINITIALIZE PARITY REGISTER
927	002454	062700	000010		3\$:	ADD	#10,R0		;MOVE POINTER TO POINT TO ADDRESS
928									;OF NEXT REGISTER
929	002460	005723				TST	(R3)+		;INCREMENT POINTER TO INDICATOR
930	002462	020027	000770			CMP	RO,#TREG		;DONE?
931	002466	002737				BLT	2\$;NO- LOOP
932									
933									
934									
935									
936									
937									
938									

MAP CORRESPONDENCE BETWEEN PARITY REGISTERS AND MEMORY, AND TYPE RESULTS
NOTE THAT IF PARITY MEMORY IS NOT LOCATED CORRECTLY BY THIS SUBTEST
IT IS DUE TO ONE OF THE FOLLOWING FAILURES:
-SETTING WRITE WRONG PARITY DID NOT CAUSE BAD PARITY TO BE WRITTEN

```

939
940
941
942
943
944
945
946
947
948
949
950
951 002470 104001
952 002472 012777 000003 176402
953 002500 005767 176060
954 002504 001044
955 002506 004767 011016
956 002512 004767 006776
957 002516 004767 007320
958
959
960 002522 005067 176344
961 002526 005067 176342
962 002532 012701 000570
963 002536 032711 000001
964 002542 001006
965 002544 056167 000002 176320
966 002552 056167 000004 176314
967 002560 062701 000010
968 002564 020127 000770
969 002570 103762
970 002572 004767 007760
971 002576 005267 175762
972 002602 032777 001000 176270
973 002610 001402
974 002612 000000
975
976 002614 000742
977
978
979
980
981
982
983
984
985 002616 104001
986 002620 012777 000004 176254
987 002626 012767 000100 013340
988 002634 004767 010670
989 002640 005767 175706
990 002644 001004
991 002646 004767 010556
992 002652 004767 010722
993
994

```

```

- PARITY GENERATE OR DETECT LOGIC FAILED
- PARITY ERROR BIT FAILED TO SET
- PARITY BITS IN MEMORY LOCATION FAILED (I.E. BIT STUCK AT GOOD PARITY VALUE)
NOTE THAT SETTING SWITCH REGISTER SWITCH 9 WILL CAUSE A HALT AFTER THE MAP
IS TYPED. IF YOU WISH TO CHANGE THE MAP TO ISOLATE THE CAUSE OF A MAPPING
FAILURE, YOU CAN DO THIS ONCE THE PROCESSOR IS HALTED. SEE THE DESCRIPTION
IN THE LISTING (PRECEDING THE MAP TAG "MPRO" AT LOCATION 600) FOR THE MEANING
OF THE MAP CONTENTS. AFTER MAKING THE DESIRED CHANGES, PRESS CONTINUE. THE NEW
MAP WILL BE TYPED AND IF SWITCH 9 IS LEFT SET THE PROCESS WILL BE REPEATED.
IF SWITCH 9 IS NOT LEFT SET THE PROGRAM WILL PROCEED TO TEST THE PARITY MEMORY
AND REGISTERS AS RECORDED IN THE NEW MAP.
*****

```

```

TEST3: SCOPE
MOV #3, @DISPLAY ; LOAD THE TEST NUMBER INTO THE DISPLAY
TST MTYFG ; IF MAPPING HAS ALREADY BEEN DONE
BNE TEST4 ; SKIP SUBTEST
JSR %7, CLRPAR ; MAP MEMORY
JSR %7, MAPMEM ; FIND PARITY MEMORY AND CORRESPONDING
JSR %7, MAPREG ; REGISTERS USING WRITE WRONG PARITY
; WITHOUT ACTION ENABLE SET
; INITIALIZE LOCATIONS INDICATING
; TOTAL PARITY MEMORY PRESENT
CONT3: CLR PMEML
CLR PMEMH
MOV #MPRO, R1
1$: BIT #1, @R1
BNE 2$
BIS 2(R1), PMEML ; FLAG EXISTING PARITY MEMORY (LOW 64K)
BIS 4(R1), PMEMH ; FLAG EXISTING PARITY MEMORY (HIGH 64K)
2$: ADD #10, R1
CMP R1, #TREG
BLO 1$
JSR %7, TMAP ; TYPE MAP
INC MTYFG ; INDICATE MAPPING DONE
BIT #BIT9, @SWR ; SWITCH 9 SET?
BEQ .+6 ; NO- BRANCH
HALT ; YES- SWITCH 9 SET INDICATING HALT
; AFTER TYPING PARITY MEMORY MAP
; GO TYPE NEW MAP TO VERIFY USER'S INTENT
BR CONT3

```

```

*****
SHOW THAT ASSERT PB WORKS CORRECTLY FOR EACH REGISTER
SHOW THAT NO TRAP OCCURS IF ACTION ENABLE (AE) IS NOT SET
SHOW THAT SETTING AE WITH ERROR ALREADY SET DOESN'T CAUSE A TRAP
NOTE THAT IF A KT11 IS PRESENT, IT IS USED DURING THIS SUBTEST
*****
TEST4: SCOPE
MOV #4, @DISPLAY ; LOAD THE TEST NUMBER INTO THE DISPLAY
MOV #100, IMAX
JSR %7, CLRPAR ; CLEAR ALL PARITY REGISTERS
TST NOKT ; KT11 PRESENT?
BNE 1$ ; NO-BRANCH
JSR %7, NRALL ; YES-MAP ALL PAGES NON RESIDENT
JSR %7, MAP1 ; THEN MAP KERNEL 0 TO BANK 0, KERNEL
; 7 TO EXTERNAL BANK, SET KERNEL
; 0, 1, AND 7 RW, AND TURN ON KT11

```


995	002656	012700	000570	1S:	MOV	#MPRO, R0	; SETUP TO FIND REGISTERS PRESENT
996	002662	012702	000772		MOV	#INDCO, R2	
997	002666	032710	000001	LOOP4:	BIT	#1, R0	; IS THIS REGISTER PRESENT?
998	002672	001405			BEQ	TST4	; YES-BRANCH TO TEST IT
999	002674	062700	000010	LOP4:	ADD	#10, R0	; NO-CHECK FOR ANOTHER ONE
1000	002700	005722			TST	(R2)+	; INCREMENT PTR9LT
1001	002702	103771			BLO	LOOP4	
1002	002704	000457			BR	DONE4	; BRANCH WHEN ALL REGISTERS HAVE BEEN TESTED
1003	002706	004767	010732	TST4:	JSR	%7, LOCATM	; LOCATE MEMORY CORRESPONDING TO
1004							; THIS REGISTER- IF NO KT11, R1 SHOULD
1005							; BE RETURNED CONTAINING THE ADDRESS
1006							; OF THE 1ST LOCATION CONTROLLED BY THIS
1007							; REGISTER (INCLUDING EXTERNAL INTERLEAVE
1008							; OFFSET IF NEEDED)
1009							; IF KT11 IS PRESENT, R1 SHOULD BE RETURNED
1010							; POINTING TO THE 1ST LOCATION CONTROLLED
1011							; BY THIS REGISTER, MAPPED THRU KERNEL
1012							; PAGE 1. KERNEL PAGE 1 SHOULD BE
1013							; MAPPED TO THE CORRECT BANK
1014	002712	032701	000001		BIT	#1, R1	; IS ERROR RETURN INDICATED?
1015	002716	001403			BEQ	.+10	; NO- BRANCH
1016	002720	104002			ERROR		; MAP INDICATES NO PARITY MEMORY
1017							; IS CONTROLLED BY THIS REGISTER
1018							; R0 POINTS TO THE ADDRESS OF THE
1019							; PARITY REGISTER
1020	002722	000167	177746		JMP	LOP4	; SETUP PARITY TRAP RETURN
1021	002726	012737	003014	000114	MOV	#TRP4A, R#114	; SET WRITE WRONG PARITY
1022	002734	012770	000004	000000	MOV	#WWP, R(R0)	; WRITE CONTENTS OF LOCATION WITH
1023	002742	011111			MOV	R1, R1	; WRONG PARITY
1024							; CLEAR PARITY REGISTER
1025	002744	005070	000000		CLR	R(R0)	; READ BAD PARITY WITH ACTION ENABLE
1026	002750	005711			TST	R1	; CLEARED- NO TRAP EXPECTED
1027							; CHANGE PARITY TRAP RETURN
1028	002752	012737	003022	000114	MOV	#TRP4B, R#PARVEC	; SET ACTION ENABLE WITH PARITY ERROR
1029	002760	052770	000001	000000	BIS	#AE, R(R0)	; ALREADY SET- SHOULDN'T TRAP YET
1030							; CHANGE PARITY TRAP RETURN
1031	002766	012737	003030	000114	MOV	#TRP4C, R#PARVEC	; READ LOCATION AGAIN- SHOULD GET
1032	002774	005711			TST	R1	; A PARITY TRAP DUE TO READING BAD
1033							; PARITY WITH ACTION ENABLE SET
1034							; NO PARITY TRAP AFTER READING LOCATION
1035	002776	104002			ERROR		; WHICH SHOULD CONTAIN BAD PARITY-
1036							; R1 CONTAINS ADDRESS OF MEMORY LOCATION
1037							; (VIRTUAL, THRU KERNEL PAGE 1, IF KT11
1038							; PRESENT). R0 POINTS TO THE ADDRESS OF
1039							; THE PARITY REGISTER IN WHICH AE WAS SET
1040							; CLEAR PARITY REGISTER
1041	003000	005070	000000	CONT4:	CLR	R(R0)	; CLEAR BAD PARITY
1042	003004	005511			ADC	R1	; CLEAR PARITY ERROR BIT
1043	003006	005070	000000		CLR	R(R0)	; GO TO TEST NEXT REGISTER
1044	003012	000730			BR	LOP4	; PARITY TRAP OCCURRED WITH ACTION
1045	003014	104002		TRP4A:	ERROR		; ENABLE CLEAR- R0 POINTS TO THE ADDRESS
1046							; OF THE PARITY REGISTER UNDER TEST
1047							; R1 CONTAINS THE ADDRESS OF THE MEMORY
1048							; UNDER TEST (VIRTUAL IF KT11 IS PRESENT)
1049							; RESTORE STACK POINTER
1050	003016	022626			CMP	(SP)+, (SP)+	


```

1107                                     ;WRONG PARITY
1108 003174 005711                       TST    @R1      ;DETECT WRONG PARITY
1109 003176 042770 000004 000000       BIC    @WMP,@(R0) ;CLEAR WWP IN PARITY REGISTER
1110 003204 011111                       MOV    @R1,@R1  ;RESTORE GOOD PARITY
1111 003206 005711                       TST    @R1      ;REREAD LOCATION
1112 003210 005770 000000               TST    @(R0)    ;READ CONTENTS OF PARITY REGISTER
1113 003214 100401                       BMI    .+4      ;BRANCH IF PARITY ERROR IS STILL SET
1114 003216 104002                       ERROR                                     ;PARITY ERROR BIT CLEARED BY READING
1115                                     ;GOOD PARITY (OR POSSIBLY WHILE DOING
1116                                     ;THE BIC TO CLEAR WWP). R0 POINTS TO
1117                                     ;THE PARITY REGISTER ADDRESS.
1118 003220 005070 000000               CLR    @(R0)    ;CLEAR THE PARITY REGISTER
1119 003224 000741                       BR     LOP5
1120 003226 005767 175320       DONE5: TST    NOKT
1121 003232 001002                       BNE   .+6
1122 003234 005037 177572       CLR    @#SRO   ;TURN OFF KT11 IF PRESENT
1123
1124
1125 *****
1126 ;SHOW THAT PARITY GENERATE AND DETECT LOGIC WORKS CORRECTLY FOR EACH BYTE
1127 ;SHOW THAT WRITE WRONG PARITY WORKS FOR HIGH AND LOW BYTES
1129 ;SHOW THAT WRITING INTO LOCATION WHEN WRITE WRONG PARITY IS NOT SET
1129 ;RESTORES GOOD PARITY
1130 *****
1131 003240 104001                       TEST6: SCOPE
1132 003242 012777 000006 175632       MOV    #6,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
1133 003250 004767 010254       JSR    %7,CLRPAR  ;CLEAR ALL PARITY REGISTERS
1134 003254 005767 175272       TST    NOKT      ;KT11 PRESENT?
1135 003260 001004                       BNE   1$        ;NO BRANCH
1136 003262 004767 010142       JSR    %7,NRALL  ;YES- MAP IT (KERNEL 0 TO BANK 0, RW;
1137 003266 004767 010306       JSR    %7,MAP1   ;KERNEL 7 TO EXTERNAL BANK, RW; KERNEL 1 RW)
1138                                     ;AND TURN IT ON
1139 003272 012700 000570       1$:  MOV    @MPRO,R0 ;SETUP TO FIND REGISTERS PRESENT
1140 003276 032710 000001       LOOP6: BIT   #1,@R0 ;IS THIS REGISTER PRESENT?
1141 003302 001406                       BEQ   T$T6      ;YES- BRANCH TO TEST IT
1142 003304 062700 000010       LOP6: ADD   #10,R0 ;NO- CHECK FOR ANOTHER ONE
1143 003310 020027 000770       CMP   R0,@TREG
1144 003314 103770                       BLO  LOOP6
1145 003316 000523                       BR   DONE6
1146                                     ;BRANCH TO DONE IF ALL REGISTERS
1147 003320 004767 010320       T$T6: JSR    %7,LOCATM ;HAVE BEEN TESTED
1148                                     ;LOCATE MEMORY CORRESPONDING TO
1149                                     ;THIS REGISTER- R1 SHOULD BE RETURNED
1150                                     ;CONTAINING THE ADDRESS OF THE FIRST
1151                                     ;LOCATION CONTROLLED BY THIS REGISTER
1152 003324 032701 000001       BIT   #1,R1     ;(VIRTUAL THRU KERNEL PAGE 1 IF KT11 PRESENT)
1153 003330 001403                       BEQ   .+10      ;IF NO MEMORY WAS FOUND TO CORRESPOND
1154 003332 104002                       ERROR          ;ODD ADDRESS IS RETURNED-BRANCH IF OK
1155                                     ;MAP INDICATES NO PARITY MEMORY IS
1156                                     ;CONTROLLED BY THIS REGISTER
1157                                     ;R0 POINTS TO THE ADDRESS OF THE
1158 003334 000167 177744       JMP   LOP6      ;PARITY REGISTER
1159                                     ;AFTER ERROR, CHECK FOR NEXT REGISTER
1160 ;FIRST SHOW THAT IF THE PARITY REGISTER IS CLEARED INITIALLY,
1161 ;PARITY ERROR DOESN'T SET
1162 003340 005011                       CLR    @R1      ;INITIALIZE LOCATION UNDER TEST

```


1275 003672 005070 000000
 1276 003676 000760
 1277 003700 012737 000116 000114 DONE7:
 1278 003706 000405 TRP7:
 1279 003710 104002
 1280
 1281
 1282
 1283
 1284 003712 022626
 1285 003714 005070 000000
 1286 003720 000747
 1287
 1288
 1289
 1290
 1291
 1292
 1293
 1234

```

CLR      2(RO)      ;CLEAR PARITY REGISTER
BR       LOOP7     ;GO CHECK NEXT REGISTER
MOV      #PARVEC+2,2#PARVEC
BR       TEST10
TRP7:    ERROR     ;TRAP OCCURRED WHEN PARITY ERROR BIT
                        ;WAS SET VIA A BIS INSTRUCTION
                        ;WITH ACTION ENABLE ALREADY SET.
                        ;RO POINTS TO THE ADDRESS OF THE
                        ;PARITY REGISTER
                        ;RESTORE STACK POINTER
CMP      (SP)+,(SP)+
CLR      2(RO)     ;CLEAR PARITY REGISTER
BR       LOOP7

```

;*****
 ;SHOW THAT REPEATED PARITY ERRORS WILL CAUSE REPEATED TRAPS IF ACTION
 ;ENABLE IS SET AND PARITY ERROR IS LEFT SET.
 ;SHOW THAT THE ERROR ADDRESS BITS (11-5) TRACK (ONLY FOR CORE PARITY REGISTERS)
 ;*****

1295	003722	104001			TEST10: SCOPE				
1296	003724	012777	000010	175150	MOV	#10, DISPLAY			: LOAD THE TEST NUMBER INTO THE DISPLAY
1297	003732	004767	007572		JSR	%7, CLRPAR			: INITIALLY CLEAR ALL PARITY REGISTERS
1298	003736	005767	174610		TST	NOKT			: KT11 PRESENT?
1299	003742	001004			BNE	1\$: NO- BRANCH
1300	003744	004767	007460		JSR	%7, NRALL			: YES- INITIALLY MAP ALL PAGES NR
1301	003750	004767	007624		JSR	%7, MAP1			: MAP KERNEL 0 TO BANK J, RW
1302									: KERNEL 7 TO THE EXTERNAL BANK, RW
1303									: MAKE KERNEL PAGE 1 RW AND TURN ON THE KT11
1304	003754	012700	000570		1\$: MOV	#MPRO, R0			
1305	003760	012702	000772		MOV	#INDCO, R2			
1306	003764	032710	000001		LUP10: BIT	#1, R0			
1307	003770	001407			BEQ	TST10			: BRANCH TO TEST REGISTER IF PRESENT
1308	003772	062700	000010		LOOP10: ADD	#10, R0			
1309	003776	005722			TST	(R2)+			
1310	004000	020027	000770		CMP	R0, #TREG			
1311	004004	103767			BLO	LUP10			
1312	004006	000507			BR	DONE10			: BRANCH IF ALL REGISTERS HAVE BEEN TESTED
1313	004010	004767	007630		TST10: JSR	%7, LOCATM			
1314	004014	032701	000001		BIT	#1, R1			: ERROR RETURN INDICATED?
1315	004020	001403			BEQ	+.10			: BRANCH IF NO
1316	004022	104002			ERROR				: MAP INDICATES THERE IS NO MEMORY
1317									: CORRESPONDING TO THIS REGISTER
1318									: R0 POINTS TO THE ADDRESS OF
1319									: THE PARITY REGISTER
1320	004024	000167	177742		JMP	LOOP10			
1321	004030	012770	000004	000000	MOV	#WWP, R0			: SET WRITE WRONG PARITY
1322	004036	011111			MOV	R1, R1			: WRITE WRONG PARITY IN FIRST LOCATION
1323	004040	016161	004000	004000	MOV	4000(R1), 4000(R1)			: WRITE WRONG PARITY IN SECOND LOCATION
1324	004046	012770	000001	000000	MOV	#AE, R0			: SET ACTION ENABLE AND CLEAR REST
1325	004054	012737	004106	000114	MOV	#TRP10, #PARVEC			: SETUP PARITY TRAP RETURN
1326	004062	012767	000010	000152	MOV	#10, COUNT			: SETUP COUNTER TO EXECUTE INSTRUCTION 1
1327									: (INST1) TEN TIMES
1328	004070	005711			INST1: TST	R1			: READ WRONG PARITY WITH AE SET- SHOULD
1329									: TRAP TO TRP10
1330	004072	104002			ERROR				: NO PARITY TRAP OCCURRED. R0 POINTS TO
1331									: ADDRESS OF THE PARITY REGISTER BEING
1332									: TESTED.
1333	004074	000441			BR	CONT10			
1334	004076	005761	004000		INST2: TST	4000(R1)			: READ WRONG PARITY FROM SECOND ADDRESS
1335									: WITH AE SET- SHOULD TRAP TO TRP10A
1336	004102	104002			ERROR				: NO PARITY TRAP OCCURRED. R0 POINTS TO
1337									: THE ADDRESS OF THE PARITY REGISTER
1338									: BEING TESTED
1339	004104	000435			BR	CONT10			
1340	004106	005367	000130		TRP10: DEC	COUNT			: HAS PARITY TRAP OCCURRED TEN TIMES?
1341	004112	001413			BEQ	1\$: YES- BRANCH
1342	004114	022712	000001		CMP	#1, (R2)			: IS THIS A CORE PAR REG?
1343	004120	001005			BNE	2\$: NO, BRANCH (NO ERROR
1344									: ADDRESS BITS FOR MOS PAR
1345	004122	032770	000040	000000	BIT	#BITS, R0			: IF ERROR ADDRESS BITS ARE TRACKING,
1346									: BIT 5 SHOULD BE CLEAR (ONLY FOR CORE PARITY)
1347	004130	001401			BEQ	+.4			
1348	004132	104002			ERROR				: PARITY ERROR ADDRESS BITS INCORRECT
1349									: R0 POINTS TO THE ADDRESS OF THE PARITY
1350									: REGISTER. R1 CONTAINS THE ADDRESS

```

1351                                     ; REFERENCED TO CAUSE A PARITY TRAP
1352                                     ; (VIRTUAL IF KT11 IS PRESENT)
1353 004134 012716 004070      2$:  MOV      #INST1, @SP      ; GO EXECUTE INSTRUCTION 1 AGAIN
1354 004140 000002                                     ;
1355 004142 012737 004156 000114 1$:  MOV      #TRP10A, @#PARVEC      ; CHANGE PARITY TRAP RETURN
1356 004150 012716 004076                                     ; GO EXECUTE INSTRUCTION 2
1357 004154 000002                                     ;
1358 004156 022712 000001      TRP10A: CMP      #1, (R2)      ; IS THIS A CORE REG?
1359 004162 001005                                     ; NO, BRANCH
1360 004164 032770 000040 000000      BNE      1$
1361                                     BIT      #BITS, @ (R0)      ; PARITY TRAP OCCURRED- CHECK PARITY
1362 004172 001001                                     ; ERROR ADDRESS BITS
1363                                     BNE      .+4      ; BRANCH IF OK (IF THE PARITY ERROR
1364 004174 104002      ERROR      ; ADDRESS BITS TRACKED, BIT 5 WILL BE SET)
1365                                     ; PARITY ERROR ADDRESS BITS INCORRECT
1366                                     ; R0 POINTS TO THE ADDRESS OF THE
1367                                     ; PARITY REGISTER. THE ADDRESS REFERENCED
1368                                     ; TO CAUSE THE ERROR WAS THAT IN
1369                                     ; R1 PLUS 4000 (OCTAL).
1369 004176 022626                                     ;
1370 004200 012737 000116 000114 1$:  CMP      (SP)+, (SP)+      ; RESTORE TRAPCATCHER
1371 004206 005070 000000      CONT10: MOV      #PARVEC+2, @#PARVEC      ; CLEAR PARITY REGISTER
1372 004212 005511                                     ; CLEAR BAD PARITY
1373 004214 005561 004000      ADC      4000(R1)
1374 004220 005070 000000      CLR      @ (R0)      ; CLEAR PARITY ERROR BIT IF SET
1375 004224 000662                                     ;
1376 004226 005767 174320      DONE10: BR      LOOP10
1377 004232 001002      TST      NOKT
1378 004234 005037 177572      BNE      .+6
1379 004240 000401      CLR      @#SRO      ; TURN OFF KT11 IF PRESENT
1380 004242 000000      BR      TEST11
1381      COUNT: 0
1382
1383
1384      ; *****
1385      ; IF MULTIPLE PARITY ERRORS OCCUR DURING ONE INSTRUCTION (WITH ACTION ENABLE
1386      ; NOT SET) THE ERROR ADDRESS BITS WILL RECORD THE LAST ERROR (ONLY FOR CORE PARITY
1387      ; REGISTERS)
1388      ; *****
1389 004244 104001      TEST11: SCOPE
1390 004246 012777 000011 174626      MOV      #11, @DISPLAY      ; LOAD THE TEST NUMBER INTO THE DISPLAY
1391 004254 004767 007250      JSR      %7, CLRPAR      ; INITIALLY CLEAR ALL PARITY REGISTERS
1392 004260 005767 174266      TST      NOKT      ; KT11 PRESENT?
1393 004264 001004      BNE      1$      ; NO- BRANCH
1394 004266 004767 007136      JSR      %7, NRALL      ; YES, MAP KERNEL PAGE 0 TO BANK 0, RW
1395 004272 004767 007302      JSR      %7, MAP1      ; MAP KERNEL PAGE 7 TO THE EXTERNAL BANK, RW
1396                                     ; SET KERNEL PAGE 1 RW AND TURN ON KT11
1397 004276 012700 000570      1$:  MOV      #MPRO, R0      ; SETUP TO GET ADDRESSES OF REGISTERS PRESENT
1398 004302 012703 000772      MOV      #INDCO, R3
1399 004306 032710 000001      LUP11: BIT      #1, @R0
1400 004312 001003      BNE      LOOP11      ; IF THIS REG NOT PRESENT, SKIP
1401 004314 022713 000001      CMP      #1, (R3)      ; IS THIS A CORE PAR REG?
1402 004320 001407      BEQ      TEST11      ; YES, THEN TEST IT
1403                                     ; IF NOT CORE, SKIP THIS REGISTER
1404 004322 062700 000010      LOOP11: ADD      #10, R0
1405 004326 005723      TST      (R3)+
1406 004330 020027 000770      CMP      R0, #TREG

```


1463	004512	012700	000570	15:	MOV	#MPRO, R0	; SETUP TO GET ADDRESSES OF REGISTERS PRESENT
1464	004516	012702	000772		MOV	#INDCO, R2	
1465	004522	032710	000001	LUP12:	BIT	#1, R0	
1466	004526	001003			BNE	LOOP12	; SKIP, IF THIS REG NOT PRESENT
1467	004530	022712	000001		CMP	#1, (R2)	; REG PRESENT, IS IT CORE?
1468	004534	001407			BEQ	TST12	; YES, DO* THIS TEST
1469							; SKIP, IF THIS REG IS NOT CORE
1470	004536	062700	000010	LOOP12:	ADD	#10, R0	
1471	004542	005722			TST	(R2)+	
1472	004544	020027	000770		CMP	R0, #TREG	
1473	004550	103764			BLO	LUP12	
1474	004552	000474			BR	DONE12	; BRANCH TO DONE IF ALL REGISTERS
1475							; HAVE BEEN TESTED
1476	004554	004767	007064	TST12:	JSR	%7, LOCATM	; LOCATE MEMORY CORRESPONDING TO THIS
1477							; REGISTER
1478	004560	032701	000001		BIT	#1, R1	; ERROR RETURN INDICATED?
1479	004564	001403			BEQ	.+10	; NO- BRANCH
1480	004566	104002			ERROR		; NO MEMORY IN MAP CORRESPONDING TO
1481							; THIS REGISTER, R0 POINTS TO
1482							; THE ADDRESS OF THE PARITY REGISTER
1483	004570	000167	177742		JMP	LOOP12	
1484	004574	012737	004630	000114	MOV	#TRP12, @#PARVEC	; SET UP PARITY TRAP RETURN
1485	004602	012770	000004	000000	MOV	#WWP, @ (R0)	; SET WRITE WRONG PARITY
1486	004610	012711	125252		MOV	#125252, @R1	; WRITE WRONG PARITY IN TEST LOCATION
1487	004614	012770	000001	000000	MOV	#AE, @ (R0)	; SET ACTION ENABLE AND CLEAR
1488							; WRITE WRONG PARITY
1489	004622	005211			INC	@R1	; DO DATIP DATO WITH ACTION ENABLE
1490							; SET- SHOULD ABORT ON DATIP AND
1491							; RESTORE ORIGINAL DATA
1492	004624	104002			ERROR		; NO ABORT OCCURRED ON READING LOCATION
1493							; WHICH SHOULD CONTAIN BAD PARITY
1494							; (WITH AE SET).
1495							; R0 POINTS TO ADDRESS OF PARITY REGISTER.
1496							; R1 CONTAINS ADDRESS OF TEST LOCATION
1497							; (VIRTUAL THRU KERNEL PAGE 1 IF KT11 IS
1498							; PRESENT)
1499	004626	000440			BR	CONT12	
1500	004630	005070	000000	TRP12:	CLR	@ (R0)	; PARITY TRAP OCCURRED- CLEAR PARITY REGISTER
1501	004634	021127	125252		CMP	@R1, #125252	; ORIGINAL DATA RESTORED?
1502	004640	001401			BEQ	.+4	; YES, BRANCH
1503	004642	104002			ERROR		; NO- DATIP WHICH GOT A PARITY ERROR
1504							; TRAP ALTERED CONTENTS OF LOCATION
1505							; READ. ADDRESS OF TEST LOCATION IS IN R1
1506							; (IF KT11 IS PRESENT, ADDRESS IN R1
1507							; IS VIRTUAL THRU KERNEL PAGE 1)
1508							; R0 POINTS TO ADDRESS OF PARITY REGISTER
1509	004644	005770	000000		TST	@ (R0)	; MAKE SURE PARITY ERROR SET WHEN
1510	004650	100401			BMI	.+4	; DATA WAS REREAD IN THE ABOVE CMP
1511	004652	104002			ERROR		; DATIP WHICH GOT A PARITY ERROR TRAP
1512							; ALTERED THE PARITY OF THE LOCATION READ
1513							; R1 CONTAINS ADDRESS OF TEST LOCATION
1514							; (VIRTUAL THRU KERNEL 1 IF KT11 PRESENT)
1515	004654	022626			CMP	(SP)+, (SP)+	; RESTORE STACK POINTER
1516	004656	012770	000004	000000	MOV	#WWP, @ (R0)	; SET WRITE WRONG PARITY AND CLEAR
1517							; PARITY ERROR
1518	004664	012711	125252		MOV	#125252, @R1	; REWRITE DATA WITH WRONG PARITY

```

1519 004670 005070 000000          CLR      2(R0)          ;CLEAR PARITY REGISTER
1520 004674 012737 000116 000114  MOV      #PARVEC+2,2#PARVEC ;RESTORE TRAPCATCHER
1521 004702 005211                INC      2R1           ;SINCE AE IS CLEAR, INSTRUCTION SHOULD
1522                                ;COMPLETE AND SHOULD CLEAR BAD PARITY
1523 004704 005070 000000          CLR      2(R0)          ;CLEAR PARITY ERROR BIT
1524 004710 022711 125253          CMP      #125253,2R1   ;CHECK DATA
1525 004714 001401                BEQ      .+4
1526 004716 104002                ERROR
1527                                ;DATIP, DATO TO A LOCATION CONTAINING BAD
1528                                ;PARITY WITHOUT AE SET LEFT INCORRECT
1529                                ;DATA. R0 POINTS TO THE ADDRESS OF
1530                                ;THE PARITY REGISTER. R1 CONTAINS THE
1531                                ;ADDRESS OF THE TEST LOCATION (VIRTUAL
1532                                ;THRU KERNEL PAGE 1 IF KT11 IS PRESENT)
1532 004720 005770 000000          TST      2(R0)          ;CHECK PARITY ERROR BIT
1533 004724 100001                BPL      .+4
1534 004726 104002                ERROR
1535                                ;DATIP, DATO WITH AE CLEAR DID
1536                                ;NOT CLEAR BAD PARITY IN LOCATION
1537                                ;ADDRESSED. R0 POINTS TO THE ADDRESS
1538                                ;OF THE PARITY REGISTER. R1 CONTAINS
1539                                ;THE ADDRESS OF THE TEST LOCATION
1540                                ;(VIRTUAL THRU KERNEL PAGE 1 IF KT11
1541                                ;IS PRESENT)
1541 004730 005070 000000          CONT12: CLR      2(R0)          ;CLEAR PARITY REGISTER
1542 004734 005011                CLR      2R1           ;CLEAR LOCATION TO RESTORE GOOD PARITY
1543 004736 005070 000000          CLR      2(R0)          ;CLEAR PARITY ERROR IF SET
1544 004742 000675                BR       LOOP12        ;GO CHECK FOR ANOTHER PARITY
1545                                ;REGISTER
1546 004744 012737 000116 000114  DONE12: MOV      #PARVEC+2,2#PARVEC ;RESTORE TRAPCATCHER
1547 004752 005767 173574          TST      NOKT
1548 004756 001002                BNE      .+6
1549 004760 005037 177572          CLR      2#SRO        ;TURN OFF KT11 IF PRESENT
1550
1551
1552
1553
1554
1555
1556
1557
1558 004764 104001                ;*****
1559 004766 012777 000013 174106  TEST13: SCOPE          ;SHOW THAT IF AN INSTRUCTION DOING A DATI (BUT NO DATO TO THE SAME LOCATION)
1560 004774 004767 006530                MOV      #13,2DISPLAY ;GETS A PARITY ERROR, THE ORIGINAL DATA IS UNALTERED, WHETHER OR NOT ACTION
1561 005000 005767 173546                JSR      %7,CLRPAR     ;ENABLE IS SET
1562 005004 001004                TST      NOKT          ;*****
1563 005006 004767 006416                BNE      1$            ;LOAD THE TEST NUMBER INTO THE DISPLAY
1564 005012 004767 006562                JSR      %7,NRALL      ;KT11 PRESENT?
1565                                ;NO- BRANCH
1566 005016 012700 000570                JSR      %7,MAP1       ;YES, MAP KERNEL 0 TO BANK 0, KERNEL
1567 005022 032710 000001                JSR      %7,MAP1       ;7 TO THE EXTERNAL BANK, AND KERNEL
1568 005026 001406                ;0, 1, AND 7 RW
1569                                ;SETUP TO GET ADDRESSES OF REGISTERS PRESENT
1570 005030 062700 000010                1$: MOV      #MPRO,R0
1571 005034 020027 000770                LUP13: BIT      #1,2R0
1572 005040 103770                BEQ      T$13          ;IF THIS REGISTER IS PRESENT, GO
1573 005042 000470                ;TEST IT
1574                                ;BRANCH IF ALL REGISTERS HAVE BEEN
1574                                ;TESTED

```



```

1631                                     ; IF KT11 IS PRESENT). R0 POINTS TO THE
1632                                     ; ADDRESS OF THE PARTIY REGISTER.
1633 005210 005070 000000          CONT13: CLR      2(R0)      ; CLEAR PARITY REGISTER
1634 005214 005011                                     CLR      2R1      ; CLEAR LOCATION
1635 005216 005070 000000          CLR      2(R0)      ; CLEAR PARITY ERROR IF SET
1636 005222 000702          BR        LOOP13      ; GO CHECK FOR ANOTHER PARITY
1637                                     ; REGISTER
1638 005224 012737 000116 000114  DONE13: MOV     #PARVEC+2,2#PARVEC ; RESTORE TRAPCATCHER
1639 005232 005767 173314          TST     NOKT
1640 005236 001002          BNE     .+6
1641 005240 005037 177572          CLR     2#SR0      ; TURN OFF KT11 IF PRESENT
1642
1643
1644
1645                                     ; *****
1646                                     ; CHECK PARITY MEMORY WITH SERIES OF PATTERNS FROM 4K TO 28K
1647                                     ; ENABLE PARITY TRAP
1648                                     ; *****
1649 005244 104001          TEST14: SCOPE
1650 005246 012777 000014 173626  MOV     #14,2DISPLAY      ; LOAD THE TEST NUMBER INTO THE DISPLAY
1651 005254 005067 010714          CLR     IMAX      ; DON'T ITERATE THE REST OF THE SUBTESTS
1652 005260 004767 006244          JSR    PC,CLRPAR      ; CLEAR ALL PARITY REGISTERS
1653 005264 012737 005566 000114  MOV     #TRAP14,2#PARVEC ; SETUP PARITY TRAP RETURN
1654 005272 012767 000002 173220  MOV     #2,BITPT      ; INITIALIZE BANK INDICATOR TO BANK 1
1655 005300 012767 020000 173210  MOV     #20000,ADRPT   ; INITIALIZE MEMORY STARTING ADDRESS
1656 005306 036767 173206 173556  LOOP14: BIT   BITPT,#MEML ; DOES THIS 4K HAVE PARITY?
1657 005314 001012          BNE     TST14      ; YES, TEST IT
1658 005316 062767 020000 173172  LUP14: ADD   #20000,ADRPT ; NO- UPDATE MEMORY ADDRESS
1659 005324 006367 173170          ASL    BITPT      ; UPDATE BIT POINTER
1660 005330 022767 000200 173162  CMP     #200,BITPT    ; THIS 28K DONE?
1661 005336 003363          BGT    LOOP14      ; NO, BRANCH TO SEE IF NEXT 4K
1662                                     ; SHOULD BE TESTED
1663 005340 000443          BR     DONE14      ; YES, EXIT
1664 005342 012704 001040          TST14: MOV   #PARPAT,R4 ; INITIALIZE PATTERN POINTER
1665 005346 016767 173144 173154  MOV     ADRPT,HIADR   ; SET UPPER LIMIT FOR THIS 4K
1666 005354 062767 020000 173146  ADD     #20000,HIADR
1667 005362 016705 173130          MOV    ADRPT,#5
1668 005366 005025          2$: CLR    (5)+      ; INITIALLY CLEAR CORE BLOCK UNDER TEST
1669 005370 020567 173134          CMP    R5,HIADR
1670 005374 103774          BLO   2$
1671 005376 012701 000570          MOV    #MPRO,R1      ; INITIALIZE TO SET AE IN ALL REGISTERS
1672 005402 032711 000001          3$: BIT   #1,2R1
1673 005406 001003          BNE   .+10
1674 005410 012771 000001 000000  MOV     #AE,2(R1)     ; SET ACTION ENABLE IF REGISTER IS PRESENT
1675 005416 062701 000010          ADD   #10,R1
1676 005422 020127 000770          CMP   R1,#TREG
1677 005426 103765          BLO   3$
1678 005430 004767 000024          4$: JSR   %7,TPCORE    ; GO TO ROUTINE TO EXERCISE THIS 4K
1679                                     ; WITH THE CURRENT PATTERN
1680 005434 005724          TST   (4)+
1681 005436 005714          TST   (4)
1682 005440 001373          BNE   4$
1683 005442 004767 006062          JSR   PC,CLRPAR      ; YES, CLEAR ALL PARITY REGISTERS
1684 005446 000723          BR    LUP14
1685 005450 012737 000116 000114  DONE14: MOV  #PARVEC+2,2#PARVEC ; RESTORE TRAP CATCHER
1686 005456 000473          BR    TEST15      ; GO TO NEXT TEST

```



```

1687
1688
1689
1690 005460 016705 173032
1691
1692 005464 011415
1693 005466 011567 173044
1694 005472 021467 173040
1695 005476 001401
1696 005500 104002
1697
1698
1699 005502 005725
1700 005504 020567 173020
1701 005510 103765
1702 005512 005067 173252
1703
1704 005516 012701 000570
1705 005522 032711 000001
1706 005526 001003
1707 005530 005771 000000
1708 005534 100406
1709 005536 062701 000010
1710 005542 020127 000770
1711 005546 103765
1712 005550 000207
1713 005552 011167 173212
1714 005556 104004
1715
1716
1717
1718 005560 004767 006446
1719
1720
1721 005564 000207
1722
1723
1724 005566 005067 173176
1725 005572 012701 000570
1726 005576 032711 000001
1727 005602 001003
1728 005604 0 3771 000000
1729 005610 100407
1730 005612 062701 000010
1731 005616 020127 000770
1732 005622 103765
1733 005624 104002
1734 005626 000405
1735
1736 005630 011167 173134
1737 005634 104004
1738
1739
1740
1741
1742 005636 004767 006370

```

```

;ROUTINE TO WRITE AND CHECK EACH LOCATION IN 4K (STARTING AT ADDRESS
;IN ADRPT) WITH VALUE POINTED TO BY R4
↑PCORE: MOV ADRPT,R5 ;SETUP R5 TO ADDRESS MEMORY
;LOCATION BEING CHECKED
1$: MOV (4),(5) ;WRITE PATTERN INTO MEMORY
MOV (5),WAS ;READ TEST LOCATION
CMP (4),WAS ;DATA OK?
BEQ .+4 ;YES- BRANCH
ERROR ;DATA INCORRECT IN LOCATION WHOSE
;ADDRESS IS IN R5. R4 POINTS TO THE
;DATA WRITTEN.
TST (5)+ ;UPDATE ADDRESS POINTER
CMP R5,HIADR ;THIS 4K DONE?
BLO 1$ ;NO, BRANCH TO TEST NEXT LOCATION
CLR TREG ;YES, DID ANY PARITY ERRORS OCCUR
;WITHOUT TRAPPING?
2$: MOV #MPRO,R1
BIT #1,(R1)
BNE .+10
TST 2(R1)
BMI 3$ ;YES- BRANCH
ADD #10,R1
CMP R1,#TREG
BLO 2$
RTS %7 ;NO, RETURN
3$: MOV 2R1,TREG ;STORE ADDRESS OF REGISTER GETTING ERROR
ERRORS ;PARITY ERROR SET (WITH AE SET) AND
;NO TRAP OCCURRED. TREG CONTAINS
;ADDRESS OF PARITY REGISTER WHICH
;HAS ERROR BIT SET.
JSR %7,PSCAN ;SCAN FOR PARITY ERRORS AND PRINT
;18 BIT ADDRESSES OF THOSE FOUND.
;AFTER REPORTING EACH ERROR CLEAR IT
RTS %7
;PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
↑TRP14: CLR TREG
MOV #MPRO,R1 ;FIND PARITY REGISTER INDICATING PARITY ERROR
1$: BIT #1,(R1)
BNE .+10
TST 2(R1)
BMI 2$ ;BRANCH IF PARITY ERROR SET
ADD #10,R1
CMP R1,#TREG
BLO 1$
ERROR 1$
BR 3$ ;PARITY TRAP TO 114 OCCURRED DURING
;TEST 14 BUT NO REGISTERS HAVE
;PARITY ERROR SET
2$: MOV 2R1,TREG ;STORE ADDRESS OF REGISTER GETTING ERROR
ERRORS ;PARITY TRAP TO 114 OCCURRED DUE TO
;PARITY ERROR WHILE EXERCISING MEMORY
;R1 POINTS TO THE ADDRESS OF THE
;PARITY REGISTER HAVING PARITY ERROR
;BIT SET
JSR %7,PSCAN ;SCAN FOR BAD PARITY AND TYPE 18 BIT ADDRESSES

```



```

1799 006106 004767 005416          JSR    PC CLRPAR          ; YES, CLEAR ALL PARITY REGISTERS
1800 006112 000716                   BR     LOP15              ; UPDATE AND CHECK NEXT 4K
1801 006114 005037 177572          CLR    @#SRO              ; TURN OFF KT11 WHEN DONE
1802 006120 012737 000116 000114  MOV    @PARVEC+2,@#PARVEC ; RESTORE TRAPCATCHER
1803 006126 000472                   BR     TEST16             ; GO TO NEXT TEST
1804
1805          ; PARITY MEMORY TEST ROUTINE USING KT11 AND TESTING MEMORY ABOVE 28K
1806          ; WRITES AND CHECKS EACH LOCATION IN 4K USING KERNEL PAGE 1 MAPPED TO CURRENT BANK
1807 006130 000240          TPCORX: NOP
1808 006132 012705 020000          MOV    #20000,R5          ; SETUP R5 TO POINT TO THE LOCATION
1809                                     ; UNDER TEST (VIRTUAL ADDRESS)
1810 006136 011415          1$:  MOV    (4),(5)         ; WRITE PATTERN
1811 006140 011567 172372          MOV    (5),WAS           ; READ TEST LOCATION
1812 006144 021467 172366          CMP    (4),WAS           ; DATA OK?
1813 006150 001401          BEQ    .+4               ; YES- BRANCH
1814 006152 104002          ERROR                    ; NO- DATA INCORRECT IN LOCATION WHOSE
1815                                     ; VIRTUAL ADDRESS IS IN R1 (GOES THRU
1816                                     ; KERNEL PAGE 1). R4 POINTS TO
1817                                     ; THE VALUE WRITTEN.
1818 006154 005725          TST    (5)+              ; UPDATE ADDRESS POINTER
1819 006156 020527 040000          CMP    R5,#40000         ; THIS 4K DONE?
1820 006162 103765          BLO    1$                ; NO, BRANCH TO TEST NEXT LOCATION
1821 006164 005067 172600          CLR    TREG              ; YES, CHECK TO SEE IF ANY PARITY
1822                                     ; ERRORS OCCURRED WITHOUT TRAPPING
1823 006170 012701 000570          MOV    #MPRO,R1
1824 006174 032711 000001          2$:  BIT    #1,(1)         ; IS THIS PARITY REGISTER PRESENT?
1825 006200 001003          BNE    .+10              ; NO, GET NEXT ONE
1826 006202 005771 000000          TST    @ (R1)            ; YES- DID ERROR SET?
1827 006206 100406          BMI    3$                ; YES- BRANCH
1828 006210 062701 000010          ADD    #10,R1            ; NO- GET NEXT REGISTER
1829 006214 020127 000770          CMP    R1,#TREG
1830 006220 103765          BLO    2$                ; NO ERRORS- EXIT
1831 006222 000207          RTS    %7
1832 006224 011167 172540          3$:  MOV    @R1,TREG      ; STORE ADDRESS OF REGISTER GETTING ERROR
1833 006230 104004          ERRORS                    ; PARITY ERROR SET (AE ALREADY SET)
1834                                     ; AND NO TRAP OR TIMEOUT OCCURRED
1835                                     ; R1 POINTS TO THE ADDRESS OF THE
1836                                     ; PARITY REGISTER
1837 006232 000207          RTS    %7
1838
1839          ; PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
1840 006234 005067 172530          TRP15: CLR    TREG
1841 006240 012701 000570          MOV    #MPRO,R1          ; LOCATE PARITY REGISTER INDICATING ERROR
1842 006244 032711 000001          1$:  BIT    #1,(R1)
1843 006250 001003          BNE    .+10
1844 006252 005771 000000          TST    @ (R1)
1845 006256 100407          BMI    2$                ; BRANCH IF PARITY ERROR IS SET
1846 006260 062701 000010          ADD    #10,R1
1847 006264 020127 000770          CMP    R1,#TREG
1848 006270 103765          BLO    1$
1849 006272 104002          ERROR                    ; TRAP TO 114 OCCURRED DURING TEST 15 BUT
1850                                     ; NO PARITY REGISTERS HAVE PARITY ERROR SET
1851 006274 000405          BR     3$
1852 006276 011167 172466          2$:  MOV    @R1,TREG      ; STORE ADDRESS OF REGISTER GETTING ERROR
1853 006302 104004          ERRORS                    ; PARITY TRAP TO 114 OCCURRED DUE TO
1854                                     ; PARITY ERROR WHILE EXERCISING MEMORY

```

```

1855
1856
1857 006304 004767 005722
1858
1859
1860
1861 006310 022626
1862 006312 000207
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874 006314 104001
1875 006316 012777 000016 172556
1876 006324 012737 006672 000114
1877 006332 012767 000002 172160
1878 006340 012767 020000 172150
1879 006346 036767 172146 172516
1880 006354 001012
1881 006356 062767 020000 172132
1882 006364 006367 172130
1883 006370 022767 000200 172122
1884 006376 003363
1885 006400 000421
1886 006402 016767 172110 172120
1887 006410 062767 020000 172112
1888 006416 004767 005106
1889 006422 016705 172070
1890 006426 005025
1891 006430 020567 172074
1892 006434 103774
1893 006436 004767 000020
1894 006442 000745
1895 006444 004767 005060
1896 006450 012737 000116 000114
1897 006456 000167 000472
1898
1899
1900
1901
1902 006462 016705 172030
1903 006466 005067 172066
1904 006472 012767 125253 172034
1905 006500 012715 125253
1906 006504 012701 000570
1907 006510 032711 000001
1908 006514 001003
1909 006516 012771 000005 000000
1910

```

```

; "TREG" CONTAINS ADDRESS OF PARITY REGISTER
; HAVING PARITY ERROR BIT SET
; SCAN MEMORY FOR BAD PARITY AND PRINT 18
; BIT ADDRESSES OF LOCATIONS FOUND
; CLEAR BAD PARITY IN EACH AFTER
; REPORTING IT
; RESTORE STACK POINTER
; RETURN (FROM JSR TO TPCORX) TO
; TEST NEXT PATTERN

;*****
; FORCE WRONG PARITY IN EACH BYTE OF PARITY MEMORY FROM 4K TO 28K
; WRITE WRONG PARITY AND READ IT BACK WITH ACTION ENABLE SET, MAKING
; SURE THAT A TRAP OCCURS. THEN WRITE GOOD PARITY AND MAKE SURE THAT
; NO TRAP OCCURS WHEN IT IS READ. MAKE SURE THAT THE ERROR ADDRESS BITS
; (PARITY REGISTER BITS 5-11) ARE CORRECT.
;*****
TEST16: SCOPE
        MOV     #16, @DISPLAY          ; LOAD THE TEST NUMBER INTO THE DISPLAY
        MOV     #TRP16, @#PARVEC      ; SET UP TRAP RETURN
        MOV     #2, BITPT             ; INIT 4K BIT POINTER TO BANK 1
        MOV     #20000, ADRPT         ; INIT MEMORY STARTING ADDRESS
1$:     BIT     BITPT, @MEML           ; DOES THIS 4K HAVE PARITY?
        BNE     3$,                   ; YES, BRANCH TO TEST IT
        ADD     #20000, ADRPT         ; NO, UPDATE MEMORY ADDRESS BY 4K
        ASL     BITPT                 ; UPDATE BIT POINTER
        CMP     #200, BITPT           ; THIS 28K DONE?
        BGT     1$,                   ; NO, CHECK NEXT 4K
        BR     DONE16                 ; YES, EXIT
3$:     MOV     ADRPT, HIADR           ; SET UPPER LIMIT THIS 4K
        ADD     #20000, HIADR
        JSR     %7, CLRPAR            ; CLEAR ALL PARITY REGISTERS
        MOV     ADRPT, R5
5$:     CLR     (5)+                  ; CLEAR BANK UNDER TEST
        CMP     R5, HIADR
        BLO     5$,
6$:     JSR     %7, WWP16             ; GO WRITE WRONG PARITY IN EACH BYTE
        BR     2$,                   ; UPDATE AND CHECK NEXT 4K
DONE16: JSR     %7, CLRPAR            ; CLEAR ALL PARITY REGISTERS IF DONE
        MOV     @PARVEC+2, @#PARVEC  ; RESTORE TRAP CATCHER
        JMP     TEST17               ; GO TO NEXT TEST

; WRITE WRONG PARITY TEST ROUTINE - TESTS EACH BYTE IN 4K
; USING SAME DATA VALUE, WRITES AND CHECKS PARITY IN WRONG STATE
; AND THEN IN CORRECT STATE TO PROVE THAT PARITY BITS TOGGLE
WWP16:  MOV     ADRPT, R5              ; SET TEST ADDRESS POINTER
        CLR     @DOFLG                ; INDICATE TESTING LOW BYTE
        MOV     #125253, @SHOBE      ; STORE DATA FOR USE BY ERROR TYPEOUT ROUTINE
WWP16A: MOV     #125253, @R5          ; INITIALIZE TEST LOCATION
        MOV     @MPRO, R1             ; SETUP TO LOAD PARITY REGISTERS
1$:     BIT     #1, (1)
        BNE     .+10
        MOV     @WWP+@AE, @ (R1)     ; SET WRITE WRONG PARITY AND ACTION
; ENABLE IF THIS PARITY REGISTER

```

```

1911                                     ; IS PRESENT
1912 006524 062701 000010          ADD    #10,R1
1913 006530 020127 000770          CMP    R1,#TREG
1914 006534 103765                   BLO   1$
1915 006536 005767 172016          TST   ODDFLG          ; WRITING HIGH BYTE?
1916 006542 100425                   BMI   2$          ; YES, BRANCH
1917 006544 112715 000253          MOV   #253,R5          ; NO, WRITE WRONG PARITY IN LOW BYTE
1918 006550 012701 000570          5$:  MOV   #MPRO,R1          ; THIS CODE CLEARS WWP BIT IN ALL
1919 006554 032711 000001          BIT   #1,(R1)          ; PARITY REGISTERS
1920 006560 001003                   BNE   .+10
1921 006562 042771 000004 000000  BIC   #WWP,2(R1)
1922 006570 062701 000010          ADD   #10,R1
1923 006574 020127 000770          CMP   R1,#TREG
1924 006600 103765                   BLO   5$+4
1925 006602 005767 171752          TST   ODDFLG          ; TESTING HIGH OR LOW BYTE?
1926 006606 100407                   BMI   6$          ; BRANCH, IF HIGH BYTE
1927 006610 142715 000377          BICB  #377,R5          ; DETECT WRONG PARITY WITH DATIP-
1928                                     ; SHOULD TRAP TO TRP16 BEFORE DOING THE DATOB
1929 006614 000407                   BR    3$
1930 006616 112765 000252 000001  2$:  MOV   #252,1(R5)          ; WRITE WRONG PARITY IN HIGH BYTE
1931 006624 000751                   BR    5$
1932 006626 142765 000377 000001  6$:  BICB  #377,1(R5)          ; DETECT WRONG PARITY WITH DATIP-
1933                                     ; SHOULD TRAP TO TRP16 BEFORE DOING THE DATOB
1934 006634 012701 000570          3$:  MOV   #MPRO,R1          ; NO TRAP OCCURRED- SETUP TO CLEAR
1935                                     ; AE AND WWP
1936 006640 032711 000001          4$:  BIT   #1,R1
1937 006644 001003                   BNE   .+10
1938 006646 042771 000005 000000  BIC   #WWP+AE,2(R1)          ; CLEAR AE AND WWP IN ALL PARITY REGISTERS
1939 006654 062701 000010          ADD   #10,R1
1940 006660 020127 000770          CMP   R1,#TREG
1941 006664 103765                   BLO   4$
1942 006666 104002          ERROR          ; ERROR, NO TRAP AFTER WRITING AND
1943                                     ; READING WRONG PARITY IN LOCATION
1944                                     ; WHOSE ADDRESS IS IN R5 (AE AND WWP
1945                                     ; WERE SET IN ALL PARITY REGISTERS)
1946                                     ; TESTING LOW BYTE IF ODDFLG IS POSITIVE
1947                                     ; TESTING HIGH BYTE IF ODDFLG IS NEGATIVE
1948                                     ; NOTE THAT AE AND WWP WERE CLEARED
1949                                     ; BEFORE TYPING THE ERROR PRINTOUT
1950 006670 000517                   BR    CN16
1951
1952                                     ; WHEN WRONG PARITY DATA IS READ BACK SHOULD ENTER HERE VIA TRAP TO 114
1953 006672 012701 000570          TRP16: MOV  #MPRO,R1          ; PARITY TRAP OCCURRED- FIRST CLEAR
1954 006676 032711 000001          TRP16A: BIT  #1,R1          ; WWP AND AE IN ALL REGISTERS
1955 006702 001003                   BNE   .+10
1956 006704 042771 000005 000000  BIC   #AE+WWP,2(R1)
1957 006712 062701 000010          ADD   #10,R1
1958 006716 020127 000770          CMP   R1,#TREG
1959 006722 103765                   BLO   TRP16A
1960 006724 012701 000570          MOV   #MPRO,R1          ; FIND THE REGISTER THAT SENSED THE ERROR
1961 006730 012703 000772          MOV   #INDCO,R3
1962 006734 005067 172030          CLR   TREG
1963 006740 032711 000001          1$:  BIT   #1,R1          ; DOES THIS CONTROL EXIST?
1964 006744 001017                   BNE   2$          ; NO, BRANCH
1965 006746 005771 000000          TST   2(R1)          ; YES- IS ERROR SET?
1966 006752 100014                   BPL   2$          ; NO, BRANCH

```

1967	006754	005767	172010		TST	TREG				
1968	006760	001401			BEQ	.+4				: YES- WAS IT SET IN ANY OTHER REGISTER ALSO?
1969	006762	104002			ERROR					: NO- BRANCH
1970										: ERROR SET IN MORE THAN ONE PARITY REGISTER
1971										: AFTER WRITING WRONG PARITY IN LOCATION
1972	006764	036761	171530	000002	BIT	BITPT,2(R1)				: WHOSE ADDRESS IS IN R5
1973										: DOES MAP INDICATE THIS PARITY REGISTER
1974	006772	001001			BNE	.+4				: CONTROLS THIS MEMORY?
1975	006774	104002			ERROR					: YES, BRANCH
1976										: PARITY REGISTER RESPONDED TO MEMORY
1977										: NOT INCLUDED IN ITS MAP
1978										: PARITY REGISTER'S ADDRESS IS POINTED
1979										: TO BY R1. ADDRESS OF LOCATION CAUSING
1980	006776	010304			MOV	R3,R4				: PARITY ERROR IS IN R5
1981	007000	011167	171764		MOV	2R1,TREG				: STORE REGISTER ADDRESS
1982	007004	062701	000010	25:	ADD	#10,R1				
1983	007010	005723			TST	(R3)+				
1984	007012	020127	000770		CMP	R1,#TREG				
1985	007016	103750			BLO	15				: BRANCH UNTIL ALL THE PARITY
1986										: REGISTERS HAVE BEEN CHECKED
1987	007020	011567	171512		MOV	(5),WAS				: SAVE DATA FROM LOCATION UNDER TEST
1988	007024	022715	125253		CMP	#125253,2R5				: DID BICB CHANGE DATA?
1989	007030	001401			BEQ	.+4				: NO, CONTINUE
1990	007032	104003			ERRORP					: DATA WAS MODIFIED BY THE BICB WHICH
1991										: GOT A PARITY ERROR TRAP- SINCE PARITY ERROR
1992										: TRAP OCCURRED, CONTENTS SHOULD NOT HAVE
1993										: BEEN MODIFIED. R5 CONTAINS ADDRESS
1994										: OF TEST LOCATION. "TREG" CONTAINS
1995										: ADDRESS OF PARITY REGISTER SENSING
1996										: ERROR
1997	007034	005767	171730		TST	TREG				: WAS PARITY ERROR SET IN ANY REGISTERS?
1998	007040	001002			BNE	35				: YES- BRANCH
1999	007042	104002			ERROR					: PARITY TRAP OCCURRED ON READING
2000										: WRONG PARITY (WITH AE SET) BUT NO
2001										: REGISTERS HAD PARITY ERROR BIT SET.
2002										: R5 CONTAINS THE ADDRESS OF THE
2003										: TEST LOCATION.
2004	007044	000420			BR	45				
2005	007046	022714	000001	35:	CMP	#1,(R4)				
2006	007052	001015			BNE	45				
2007	007054	017701	171710		MOV	2TREG,R1				: GET PARITY REGISTER CONTENTS
2008	007060	042701	170037		BIC	#170037,R1				: MASK OFF ALL BUT ERROR ADDRESS BITS
2009	007064	010502			MOV	R5,R2				: GET ADDRESS OF LOCATION UNDER TEST
2010	007066	042702	003777		BIC	#3777,R2				: POSITION BITS IN R2
2011	007072	000302			SWAB	R2				
2012	007074	006302			ASL	R2				
2013	007076	006302			ASL	R2				
2014	007100	020122			CMP	R1,R2				: PARITY ERROR ADDRESS BITS CORRECT?
2015	007102	001401			BEQ	.+4				
2016	007104	104003			ERRORP					: ERROR ADDRESS BITS (PARITY REGISTER
2017										: BITS 11-5) ARE INCORRECT. R5 CONTAINS
2018										: THE ADDRESS OF THE TEST LOCATION.
2019										: "TREG" CONTAINS THE ADDRESS OF THE
2020										: PARITY REGISTER DETECTING THE ERROR
2021	007106	022626		45:	CMP	(SP)+,(SP)+				: RESTORE STACK POINTER
2022	007110	011515		CNT16:	MOV	(5),(5)				: RESTORE TEST LOCATION TO FIX BAD PARITY


```

2079 007342 005025          55:   CLR      (5)+          ;CLEAR BANK UNDER TEST
2080 007344 020527 040000          CMP      R5,#40000
2081 007350 103774          BLO      55
2082 007352 004767 000020          65:   JSR      %7,WWP17          ;GO WRITE WRONG PARITY AND CHECK IT
2083 007356 000742          BR       25          ;UPDATE AND CHECK NEXT 4K
2084 007360 005037 177572          DONE17: CLR      @SR0          ;TURN OFF KT11 WHEN DONE
2085 007364 012737 000116 000114          MOV      @PARVEC+2,@PARVEC          ;RESTORE TRAP CATCHER
2086 007372 000167 000436          JMP      XFR1          ;GO TO SETUP FOR NEXT TEST
2087
2088          ;WRITE WRONG PARITY TEST ROUTINE TO TEST MEMORY ABOVE 28K
2089 007376 012705 020000          WWP17:  MOV      @20000,R5          ;SET TEST ADDRESS POINTER
2090 007402 005067 171152          CLR      00DFLG          ;CLEAR FLAG TO INDICATE TESTING LOW BYTE
2091 007406 012767 125253 171120          MOV      @125253,SHDBE          ;STORE DATA FOR USE BY ERROR TYPEOUT ROUTINE
2092 007414 012715 125253          WWP17A: MOV      @125253,@R5          ;INITIALIZE LOCATION
2093 007420 012701 000570          MOV      @MPRO,R1          ;INITIALIZE REGISTER ADDRESS POINTER
2094 007424 032711 000001          15:   BIT      @1,(1)          ;DOES THIS CONTROL EXIST?
2095 007430 001003          BNE      .+10          ;NO, GET NEXT
2096 007432 012771 000005 000000          MOV      @WWP+AE,@(R1)          ;YES- SET WRITE WRONG PARITY
2097          ;AND ACTION ENABLE
2098 007440 062701 000010          ADD      #10,R1
2099 007444 020127 000770          CMP      R1,@TREG          ;ALL REGISTERS SETUP?
2100 007450 103765          BLO      15          ;NO- LOOP
2101 007452 005767 171102          TST      00DFLG          ;YES- TESTING HIGH BYTE?
2102 007456 100405          BMI      25          ;YES, BRANCH
2103 007460 112715 000253          MOV      @253,@R5          ;NO, WRITE WRONG PARITY IN LOW BYTE
2104 007464 142715 000377          BIC      @377,@R5          ;DETECT WRONG PARITY WITH DATIP-
2105          ;SHOULD TRAP TO TRP17 BEFORE DOING THE DATOB
2106 007470 000406          BR       35
2107 007472 112765 000252 000001 25:   MOV      @252,1(R5)          ;WRITE WRONG PARITY IN HIGH BYTE
2108 007500 142765 000377 000001          BIC      @377,1(R5)          ;DETECT WRONG PARITY WITH DATIP
2109          ;SHOULD TRAP TO TRP17 BEFORE DOING THE DATOB
2110 007506 012701 000570          35:   MOV      @MPRO,R1          ;IF NO TRAP, CLEAR AE AND WWP IN ALL
2111 007512 032711 000001          45:   BIT      @1,@R1          ;PARITY REGISTERS
2112 007516 001003          BNE      .+10
2113 007520 042771 000005 000000          BIC      @AE+WWP,@(R1)
2114 007526 062701 000010          ADD      #10,R1
2115 007532 020127 000770          CMP      R1,@TREG
2116 007536 103765          BLO      45
2117 007540 104002          ERROR          ;NO TRAP AFTER WRITING AND READING
2118          ;WRONG PARITY WITH AE SET- VIRTUAL
2119          ;ADDRESS OF LOCATION IS IN R5
2120          ;(MAPPED THRU KERNEL PAGE 1)
2121          ;WROTE LOW BYTE IF 00DFLG IS POSITIVE
2122          ;WROTE HIGH BYTE IF IT IS NEGATIVE
2123          ;NOTE THAT WWP AND AE WERE CLEARED
2124          ;BEFORE ERROR PRINTOUT
2125 007542 000512          BR       CNT17
2126
2127          ;WHEN WRONG PARITY DATA IS READ BACK, SHOULD ENTER HERE VIA TRAP TO 114
2128 007544 012701 000570          TRP17:  MOV      @MPRO,R1          ;PARITY TRAP OCCURRED- BEFORE CHECKING
2129 007550 032711 000001          TRP17A: BIT      @1,@R1          ;IT, CLEAR WWP AND AE IN ALL REGISTERS
2130 007554 001003          BNE      .+10
2131 007556 042771 000005 000000          BIC      @AE+WWP,@(R1)
2132 007564 062701 000010          ADD      #10,R1
2133 007570 020127 000770          CMP      R1,@TREG
2134 007574 103765          BLO      TRP17A

```


2135	007576	012701	000570		MOV	#MPRO,R1		;FIND REGISTER THAT SENSED THE ERROR
2136	007602	012703	000772		MOV	#INDCO,R3		;SETUP PTR TO INDICATOR
2137	007606	005067	171156		CLR	TREG		
2138	007612	032711	000001	1\$:	BIT	#1,R1		;DOES THIS CONTROL EXIST?
2139	007616	001017			BNE	2\$;NO BRANCH
2140	007620	005771	000000		TST	2(R1)		;YES- IS ERROR SET?
2141	007624	100014			BPL	2\$;NO BRANCH
2142	007626	005767	171136		TST	TREG		;YES- WAS IT SET IN ANY OTHER REGISTER ALSO?
2143	007632	071401			BEQ	.+4		;NO BRANCH
2144	007634	104002			ERROR			;ERROR SET IN MORE THAN ONE PARITY REGISTER
2145								;AFTER READING WRONG PARITY IN LOCATION
2146								;WHOSE VIRTUAL ADDRESS IS IN RS
2147	007636	036761	170656	000002	BIT	BITPT,2(R1)		;DOES MAP INDICATE THAT THIS REGISTER
2148								;CONTROLS THIS MEMORY?
2149		007642			INX17=.	-2		
2150	007644	001001			BNE	.+4		;YES- BRANCH
2151	007646	104002			ERROR			;PARITY REGISTER RESPONDED TO MEMORY
2152								;NOT INCLUDED IN ITS MAP. R1 POINTS TO
2153								;THE PARITY REGISTER'S ADDRESS. RS
2154								;CONTAINS VIRTUAL ADDRESS OF THE LOCATION
2155								;BEING TESTED (MAPPED THRU KERNEL
2156								;PAGE 1)
2157	007650	011167	171114		MOV	R1,TREG		;STORE REGISTER ADDRESS
2158	007654	010304			MOV	R3,R4		;STORE PTR TO INDICATOR
2159	007656	062701	000010	2\$:	ADD	#10,R1		
2160	007662	005723			TST	(R3)+		;INCREMENT PTR TO INDICATOR
2161	007664	020127	000770		CMP	R1,#TREG		
2162	007670	103750			BLO	1\$;LOOP UNTIL ALL THE PARITY REGISTERS
2163								;HAVE BEEN CHECKED
2164	007672	011567	170640		MOV	(5),R5		;SAVE CONTENTS OF LOCATION UNDER TEST
2165	007676	022715	125253		CMP	#125253,R5		;DID BICB CHANGE DATA?
2166	007702	001401			BEQ	.+4		;NO CONTINUE
2167	007704	104002			ERROR			;DATA WAS MODIFIED BY THE BICB WHICH
2168								;GOT A PARITY ERROR TRAP- SINCE PARITY
2169								;TRAP OCCURRED, CONTENTS SHOULD NOT
2170								;HAVE BEEN MODIFIED. RS CONTAINS TEST
2171								;LOCATION ADDRESS (VIRTUAL, MAPPED THRU
2172								;KERNEL PAGE 1). "TREG" CONTAINS ADDRESS OF
2173								;PARITY REGISTER DETECTING PARITY ERROR
2174	007706	005767	171056		TST	TREG		;WAS PARITY ERROR SET IN ANY REGISTER?
2175	007712	001003			BNE	3\$;YES- BRANCH
2176	007714	104002			ERROR			;PARITY TRAP OCCURRED ON READING
2177								;WRONG PARITY WITH RE SET BUT NO
2178								;REGISTER HAS THE PARITY ERROR BIT
2179								;SET. RS CONTAINS VIRTUAL ADDRESS
2180								;OF THE TEST LOCATION (MAPPED THRU
2181								;KERNEL PAGE 1)
2182	007716	000423			BR	4\$		
2183	007720	001022			BNE	4\$;NO BRANCH. DO NOT CHECK THE
2184	007722	022714	000001	3\$:	CMP	#1,(R4)		;IS THIS A CORE PARITY REGISTER?
2185	007726	001017			BNE	4\$;NO BRANCH. DO NOT CHECK THE
2186								;ADDRESS BITS OF THE LAST PARITY
2187								;ERROR
2188								;IF CORE REG, CHECK IF ADDRESS
2189								;BITS OF LAST PAR ERR WERE STORED
2190	007730	017701	171034		MOV	R2TREG,R1		;GET CONTENTS OF REGISTER DETECTING PARITY ERROR

2191	007734	042701	170037	BIC	#170037,R1	;MASK OFF ALL BUT ADDRESS BITS
2192	007740	010502		MOV	R5,R2	;CALCULATE TOP 7 BITS OF ERROR ADDRESS
2193	007742	042702	163777	BIC	#163777,R2	
2194	007746	000302		SWAB	R2	
2195	007750	006302		ASL	R2	
2196	007752	006302		ASL	R2	
2197	007754	067702	171266	ADD	2KPAR1,R2	
2198	007760	020102		CMP	R1,R2	;PARITY ERROR ADDRESS BITS CORRECT?
2199	007762	001401		BEQ	.+4	
2200	007764	104004		ERRORS		;PARITY ERROR ADDRESS BITS (PARITY REGISTER BITS 11-5) INCORRECT
2201						; "TREG" CONTAINS ADDRESS OF PARITY REGISTER. R5 CONTAINS THE VIRTUAL ADDRESS OF THE TEST LOCATION (MAPPED THRU KERNEL PAGE 1)
2202						; RESTORE STACK POINTER
2203						; RESTORE TEST LOCATION TO FIX BAD PARITY
2204						; CLEAR ERROR BIT IN THE PARITY REGISTER
2205						; READ LOCATION TO SEE IF PARITY IS GOOD
2206	007766	022626		4S: CMP	(SP)+,(SP)+	; IS PARITY ERROR SET?
2207	007770	011515		CNT17: MOV	(5),(5)	; NO BRANCH
2208	007772	005077	170772	CLR	2TREG	; WRITING LOCATION WITH WRITE WRONG
2209	007776	005715		TST	2R5	; PARITY CLEAR DIDN'T CLEAR BAD PARITY
2210	010000	005777	170764	TST	2TREG	; "TREG" CONTAINS THE ADDRESS OF THE PARITY REGISTER. R5 CONTAINS THE VIRTUAL ADDRESS OF THE TEST LOCATION (MAPPED THRU KERNEL PAGE 1)
2211	010004	100001		BPL	.+4	; TOGGLE BYTE INDICATOR
2212	010006	104002		ERROR		; BRANCH IF HIGH BYTE NOT YET TESTED
2213						; UPDATE ADDRESS POINTER
2214						; THIS 4K DONE?
2215						; NO TEST NEXT LOCATION
2216						; YES, RETURN TO CHECK FOR NEXT
2217						; BANK TO BE TESTED
2218	010010	005167	170544	COM	ODDFLG	; GO AND TEST NEXT LOCATION
2219	010014	100401		BMI	.+4	
2220	010016	005725		TST	(5)+	
2221	010020	020527	040000	CMP	R5,#40000	
2222	010024	103401		BLO	1\$	
2223	010026	000207		RTS	%7	
2224						
2225	010030	000167	177360	1S: JMP	WWP17A	
2226	010034	104001		XFR1: SCUPF		
2227						; IF THE FIRST BANK (BANK 0) IS PARITY MEMORY, SETUP TO TEST IT
2228						; COPY THE FIRST 4K TO THE SECOND 4K, MOVE THE STACK POINTER TO BANK 1,
2229						; AND THEN JUMP TO THE COPY OF TEST20 IN BANK 1
2230	010036	005737	000042	TST	2#42	
2231	010042	001402		BEQ	.+6	
2232	010044	000167	001324	JMP	DONE	; IF THE PROGRAM IS RUNNING UNDER ACT THEN
2233						; GO TO DONE
2234	010050	032767	000001 171014	BIT	#1,PMEML	
2235	010056	001002		BNE	.+6	; BRANCH IF YES
2236	010060	000167	001310	JMP	DONE	; NO- DONE WITH TEST
2237	010064	032767	000002 170774	BIT	#2,MEML	; IS THERE A SECOND 4K (BANK 1)?
2238	010072	001002		BNE	.+6	; YES, BRANCH
2239	010074	000167	001274	JMP	DONE	; NO, EXIT
2240	010100	005037	177776	CLR	2#PS	; CLEAR STATUS REGISTER
2241	010104	004767	003420	JSR	%7,CLRPAR	; CLEAR ALL PARITY REGISTERS
2242	010110	012700	010000	MOV	#10000,R0	; R0 IS COUNTER TO MOVE 4K
2243	010114	005001		CLR	R1	; R1 POINTS TO LOCATION IN BANK 0
2244	010116	011161	020000	1S: MOV	2R1,20000(R1)	; COPY FROM BANK 0 TO BANK 1
2245	010122	005721		TST	(R1)+	; MOVE POINTER
2246	010124	005300		DEC	R0	; DONE WITH 4K?


```

2294          : PARITY MEMORY TEST ROUTINE
2295          : WRITES AND CHECKS EACH LOCATION IN BANK 0 (EXCEPT 114 AND 116)
2296          : WITH VALUE POINTED TO BY R4
2297 010350 005005          CKBK0: CLR      R5          : SET ADDRESS POINTER
2298 010352 011415          1$:  MOV     (4), (5)       : WRITE PATTERN
2299 010354 021415          :      CMP     (4), (5)       : DATA OK?
2300 010356 001404          :      BEQ     +12           : YES- BRANCH
2301 010360 013746 177776  :      MOV     @#PS, -(SP)    : SETUP TO DO ERROR CALL VIA JSR
2302 010364 004767 002710  :      JSR     PC, ERR        : ERROR- DATA INCORRECT IN LOCATION
2303          :                               : WHOSE ADDRESS IS IN R5. R4 POINTS
2304          :                               : TO THE VALUE WRITTEN.
2305 010370 005725          :      TST     (5)+          : UPDATE ADDRESS POINTER
2306 010372 020527 000114  :      CMP     R5, #114      : DON'T CHANGE CONTENTS OF 114 AND 116
2307 010376 001002          :      BNE     +6            :
2308 010400 062705 000004  :      ADD     #4, R5        :
2309 010404 020527 020000  :      CMP     R5, #20000    : HAS THE WHOLE BANK BEEN TESTED WITH
2310          :                               : THIS PATTERN?
2311 010410 103760          :      BLO     1$           : NO, BRANCH TO TEST NEXT LOCATION
2312 010412 005067 170352  :      CLR     TREG          : YES, DID ANY PARITY ERROR BITS SET?
2313 010416 012701 020570  :      MOV     @MPRO+20000, R1
2314 010422 032711 000001  : 2$:  BIT     #1, (1)       :
2315 010426 001003          :      BNE     +10          :
2316 010430 005771 000000  :      TST     @ (R1)        :
2317 010434 100406          :      BMI     3$           : YES- BRANCH
2318 010436 062701 000010  :      ADD     #10, R1       :
2319 010442 020127 020770  :      CMP     R1, @TREG+20000
2320 010446 103765          :      BLO     2$           :
2321 010450 000207          :      RTS     %7           : NO- RETURN
2322 010452 013746 177776  : 3$:  MOV     @#PS, -(SP)    : SETUP TO DO ERROR CALL VIA JSR
2323 010456 004767 002616  :      JSR     PC, ERR        : ERROR- PARITY ERROR BIT SET AND NO
2324          :                               : PARITY TRAP OCCURRED
2325          :                               : (AE WAS SET) - R1 POINTS TO ADDRESS
2326          :                               : OF PARITY REGISTER
2327 010462 022626          :      CMP     (SP)+, (SP)+  : RESTORE STACK POINTER
2328 010464 000167 177644  :      JMP     DONE20        :
2329          :
2330          : PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
2331 010470 005067 170274  : TRP20: CLR     TREG        :
2332 010474 012701 020570  :      MOV     @MPRO+20000, R1 : FIND THE REGISTER RECORDING A PARITY ERROR
2333 010500 032711 000001  : 1$:  BIT     #1, (R1)       :
2334 010504 001003          :      BNE     +10          :
2335 010506 005771 000000  :      TST     @ (R1)        :
2336 010512 100412          :      BMI     2$           : BRANCH IF ERROR IS SET
2337 010514 062701 000010  :      ADD     #10, R1       :
2338 010520 020127 020770  :      CMP     R1, @TREG+20000
2339 010524 103765          :      BLO     1$           :
2340 010526 013746 177776  :      MOV     @#PS, -(SP)    : SETUP TO DO ERROR CALL VIA A JSR
2341 010532 004767 002542  :      JSR     PC, ERR        : ERROR- PARITY TRAP TO 114 OCCURRED DURING
2342          :                               : TEST 20 BUT NO REGISTERS HAVE PARITY
2343 010536 000430          :      BR     5$           : ERROR SET
2344 010540 017102 000000  : 2$:  MOV     @ (R1), R2     : SAVE CONTENTS OF PARITY REGISTER
2345 010544 005003          :      CLR     R3           : INITIALIZE ADDRESS POINTER
2346 010546 005071 000000  : 3$:  CLR     @ (R1)        : CLEAR PARITY REGISTER
2347 010552 005713          :      TST     @R3          : READ LOCATION
2348 010554 005771 000000  :      TST     @ (R1)        : PARITY ERROR SET?
2349 010560 100413          :      BMI     4$           : YES- BRANCH

```

```

2350 010562 005723          TST      (R3)+          ;MOVE POINTER
2351 010564 020327 020000  CMP      R3,#20000
2352 010570 103766          BLO     3$
2353 010572 010271 000000  MOV      R2,(R1)      ;RESTORE PARITY REGISTER CONTENTS
2354 010576 013746 177776  MOV      @#PS,-(SP)   ;SETUP TO CALL ERROR VIA JSR
2355 010602 004767 002472  JSR     PC,EAR        ;PARITY ERROR OCCURRED WHILE TESTING
2356                                     ;BANK 0 BUT NO BAD PARITY WAS FOUND
2357                                     ;DURING SCAN OF BANK 0. R1 POINTS
2358                                     ;TO THE ADDRESS OF THE PARITY REGISTER
2359                                     ;DETECTING THE ERROR
2360 010606 000404          BR      5$
2361 010610 013746 177776  4$:    MOV      @#PS,-(SP) ;SETUP TO DO ERROR CALL VIA JSR
2362 010614 004767 002460  JSR     PC,EAR        ;PARITY ERROR WHILE EXERCISING MEMORY
2363                                     ;BANK 0- R1 POINTS TO ADDRESS OF PARITY
2364                                     ;REGISTER DETECTING THE ERROR. R3 CONTAINS
2365                                     ;THE ADDRESS OF THE LOCATION HAVING
2366                                     ;BAD PARITY
2367 010620 022626          5$:    CMP      (SP)+,(SP)+
2368 010622 000207          RTS     %7           ;RETURN TO CHECK USING THE NEXT PATTERN
2369
2370
2371
2372                                     ;*****
2373                                     ;FORCE WRONG PARITY IN EACH LOCATION IN BANK 0
2374                                     ;NOTE THAT THIS SUBTEST IS EXECUTED IN BANK 1 (20000 ABOVE ADDRESSES
2375                                     ;IN THE LISTING). MAKE SURE THAT WRONG PARITY IN EACH BYTE CAN BE DETECTED,
2376                                     ;AND THAT WHEN GOOD PARITY IS WRITTEN AND READ NO PARITY ERROR IS DETECTED.
2377                                     ;CHECK ERROR ADDRESS BITS (PARITY REGISTER BITS 5-11)
2378                                     ;*****
2379 010624 013746 177776  TEST21: MOV     @#PS,-(SP) ;SAME AS SCOPE WITHOUT DOING EMT
2380 010630 004767 005240  JSR     PC,SCOPE
2381 010634 012777 000021 170240  MOV     #21,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
2382 010642 004767 002662  JSR     PC,CLRPAR    ;CLEAR ALL PARITY REGISTERS
2383 010646 005005          CLR     R5
2384 010650 005025          1$:    CLR     (R5)+      ;CLEAR PARITY ERRORS IN BANK 0
2385 010652 020527 020000  CMP     R5,#20000
2386 010656 103774          BLO     1$
2387
2388                                     ;WRITE WRONG PARITY TEST ROUTINE
2389                                     ;USING SAME DATA VALUE, WRITES AND CHECKS PARITY IN WRONG STATE
2390                                     ;AND THEN IN CORRECT STATE TO PROVE THAT PARITY BITS TOGGLE
2391 010660 005005          WWP21: CLR     R5          ;SET TEST ADDRESS POINTER
2392 010662 005067 167672  CLR     ODDFLG       ;CLEAR FLAG TO INDICATE TESTING LOW BYTE
2393 010666 012767 125253 167640  MOV     #125253,SHDBE ;STORE DATA FOR USE BY ERROR TYPEOUT ROUTINE
2394 010674 012715 125253  WWP21A: MOV     #125253,@R5 ;INITIALIZE LOCATION
2395 010700 012701 020570  MOV     @MPRO+20000,R1 ;SETUP TO SET WRITE WRONG PARITY
2396                                     ;IN ALL REGISTERS
2397 010704 032711 000001  1$:    BIT     #1(1)
2398 010710 001003          BNE     .+10
2399 010712 012771 000004 000000  MOV     @WWP,(R1)    ;SET WRITE WRONG PARITY
2400 010720 062701 000010  ADD     #10,R1
2401 010724 020127 020770  CMP     R1,@TREG+20000
2402 010730 103765          BLO     1$
2403 010732 005767 167622  TST     ODDFLG       ;TESTING HIGH BYTE?
2404 010736 100404          BMI     2$          ;YES, BRANCH
2405 010740 112715 000253  MOVB   #253,@R5     ;NO, WRITE WRONG PARITY IN LOW BYTE

```

2406	010744	005715			TST	(R5)		; DETECT WRONG PARITY WITH DATI
2407	010746	000405			BR	3\$		
2408	010750	112765	000252	00J001	2\$:	MOV	#252,1(R5)	; WRITE WRONG PARITY IN HIGH BYTE
2409	010756	105765	000001			TST	1(R5)	; DETECT WRONG PARITY WITH DATI
2410	010762	012701	020570		3\$:	MOV	#MPRO+20000,R1	; SETUP TO CLEAR WWP IN ALL PARITY REGISTERS
2411	010766	012703	000772			MOV	#INDCO,R3	; POINTER TO INDICATOR
2412	010772	032711	000001		4\$:	BIT	#1,R1	
2413	010776	001003				BNE	+10	
2414	011000	042771	000004	000000		BIC	#WWP,2(R1)	; CLEAR WWP IN ALL PARITY REGISTERS
2415	011006	062701	000010			ADD	#10,R1	
2416	011012	020127	020770			CMP	R1,#TREG+20000	
2417	011016	103765				BLC	4\$	
2418	011020	012701	020570			MOV	#MPRO+20000,R1	; CHECK TO SEE IF ANY REGISTER DETECTED
2419	011024	005067	167740			CLR	TREG	; THE PARITY ERROR
2420	011030	032711	000001		LOOP21:	BIT	#1,R1	
2421	011034	001024				BNE	5\$	
2422	011036	005771	000000			TST	2(R1)	; CHECK PARITY REGISTER
2423	011042	100021				BPL	5\$; BRANCH IF PARITY ERROR IS NOT SET
2424	011044	005767	167720			TST	TREG	; WAS PARITY ERROR SET IN ANY OTHER REGISTER?
2425	011050	001404				BEQ	+12	; NO- BRANCH
2426	011052	013746	177776			MOV	2#PS, -(SP)	; SETUP TO DO ERROR CALL VIA A JSR
2427	011056	004767	002216			JSR	PC,ERR	; ERROR- PARITY ERROR SET IN MORE THAN
2428								; ONE REGISTER AFTER WRITING WRONG PARITY
2429								; IN LOCATION WHOSE ADDRESS IS IN R5
2430	011062	032761	000001	000002		BIT	#1,2(R1)	; DOES MAP INDICATE THAT THIS REGISTER
2431								; CONTROLS THIS MEMORY?
2432	011070	001004				BNE	+12	; YES, BRANCH
2433	011072	013746	177776			MOV	2#PS, -(SP)	; NO- SETUP TO DO ERROR CALL VIA A JSR
2434	011076	004767	002176			JSR	PC,ERR	; ERROR- PARITY REGISTER RESPONDED
2435								; TO MEMORY WHICH IS NOT IN ITS MAP.
2436								; R1 POINTS TO PARITY REGISTER'S ADDRESS.
2437								; R5 CONTAINS ADDRESS OF LOCATION BEING
2438								; TESTED
2439	011102	011167	167662			MOV	2R1, TREG	; STORE REGISTER ADDRESS
2440	011106	062701	000010		5\$:	ADD	#10,R1	
2441	011112	005723				TST	(R3)+	
2442	011114	020127	020770			CMP	R1,#TREG+20000	
2443	011120	103743				BLO	LOOP21	; BRANCH UNTIL ALL REGISTERS HAVE BEEN
2444								; CHECKED
2445	011122	005767	167642			TST	TREG	; WAS PARITY ERROR SET IN ANY REGISTER?
2446	011126	001005				BNE	6\$; YES- BRANCH
2447	011130	013746	177776			MOV	2#PS, -(SP)	; NO- SETUP TO DO ERROR CALL VIA A JSR
2448	011134	004767	002140			JSR	PC,ERR	; ERROR- NO REGISTER HAS PARITY ERROR
2449								; SET AFTER READING WRONG PARITY IN LOCATION
2450								; WHOSE ADDRESS IS IN R5
2451	011140	000423				BR	7\$	
2452	011142	022713	000001		6\$:	CMP	#1,(R3)	; IS THIS A CORE PAR REG?
2453	011146	001020				BNE	7\$; NO, BRANCH(MOS PAR REG DOES NOT
2454								; STORE ADDR BITS OF PAR ERROR)
2455	011150	017701	167614			MOV	2TREG,R1	; GET PARITY REGISTER CONTENTS
2456	011154	042701	170037			BIC	#170037,R1	; MASK ALL BUT ERROR ADDRESS BITS
2457	011160	010502				MOV	R5,R2	; GET ADDRESS OF LOCATION UNDER TEST
2458	011162	042702	003777			BIC	#3777,R2	; POSITION BITS IN R2
2459	011166	000302				SWAB	R2	
2460	011170	006302				ASL	R2	
2461	011172	006302				ASL	R2	

```

2462 011174 020102          CMP      R1,R2          ;PARITY ERROR ADDRESS BITS CORRECT?
2463 011176 001404          BEQ      7$            ;BRANCH IF YES
2464 011200 013746 177776      MOV      @#PS, -(SP)   ;NO- SETUP TO DO ERROR CALL VIA A JSR
2465 011204 004767 002070      JSR      PC,ERR        ;ERROR- ADDRESS BITS (PARITY REGISTER
2466                                     ;BITS 5-11) INCORRECT- ADDRESS OF PARITY
2467                                     ;REGISTER IS CONTAINED IN LOCATION "TREG"
2468                                     ;ADDRESS OF TEST LOCATION IS IN R5
2469 011210 011515          7$:      MOV      @R5,@R5      ;RESTORE TEST LOCATION TO FIX BAD PARITY
2470 011212 005077 167552      CLR      @TREG        ;CLEAR ERROR BIT IN PARITY REGISTER
2471 011216 005715          TST      @R5          ;READ LOCATION TO SET IF PARITY IS GOOD
2472 011220 005777 167544      TST      @TREG        ;CHECK PARITY ERROR BIT
2473 011224 100004          BPL      .+12         ;BRANCH IF NOT SET
2474 011226 013746 177776      MOV      @#PS, -(SP)   ;SETUP TO DO ERROR CALL VIA A JSR
2475 011232 004767 002042      JSR      PC,ERR        ;ERROR- WRITING LOCATION WITH WRITE
2476                                     ;WRONG PARITY CLEAR DIDN'T CLEAR BAD
2477                                     ;PARITY (ADDRESS OF LOCATION IS IN R5)
2478                                     ;"TREG" +20000 CONTAINS THE ADDRESS
2479                                     ;OF THE PARITY REGISTER
2480 011236 005167 167316      COM      00DFLG        ;TOGGLE BYTE INDICATOR
2481 011242 100401          BMI      .+4          ;BRANCH IF HIGH BYTE NOT YET TESTED
2482 011244 005725          TST      (R5)+        ;UPDATE ADDRESS POINTER
2483 011246 020527 020000      CMP      R5,#20000    ;THIS 4K DONE?
2484 011252 103610          BLO      WWP21A        ;LOOP TILL ALL 4K HAS BEEN TESTED
2485 011254 004767 002250      JSR      PC,CLRPAR    ;CLEAR ALL PARITY REGISTERS
2486 011260 013746 177776      MOV      @#PS, -(SP)   ;SETUP TO CALL SCOPE VIA JSR
2487 011264 004767 004604      JSR      PC,SCOPEC    ;SCOPE
2488
2489                                     ;COPY SECOND 4K BANK BACK TO FIRST 4K AND RETURN TO FIRST 4K BANK
2490 011270 012700 010000      XFR2:   MOV      #10000,R0 ;R0 IS USED AS A COUNTER
2491 011274 005001          CLR      R1           ;R1 POINTS TO THE CURRENT LOCATION
2492 011276 016111 020000      1$:     MOV      20000(R1),@R1 ;COPY BANK 1 TO BANK 0
2493 011302 005721          TST      (R1)+
2494 011304 005300          DEC      R0
2495 011306 001373          BNE      1$
2496 011310 162706 020000      SUB      #20000,SP     ;RESTORE STACK POINTER
2497 011314 162737 020000 013350  SUB      #20000,@#ERRA1
2498 011322 162737 020000 013352  SUB      #20000,@#ERRA2
2499 011330 162737 020000 013362  SUB      #20000,@#ERRA3
2500 011336 162737 020000 013364  SUB      #20000,@#ERRA4
2501 011344 162737 020000 013374  SUB      #20000,@#ERRA5
2502 011352 022737 177570 001100  CMP      #177570,@#SWR ;DOES THE CONSOLE HAVE A SWITCH REG. ?
2503 011360 001403          BEQ      2$            ;IF SO THEN GO TO 2$
2504 011362 162737 020000 001100  SUB      #20000,@#SWR ;OTHERWISE RESTORE THE ADDRESS OF SOFTWARE
2505                                     ;SWITCH REGISTER
2506 011370 000137 011374      2$:     JMP      @#DONE        ;RETURN TO BANK 0
2507
2508
2509
2510
2511                                     ;*****
2512                                     ;AT THIS POINT EXECUTION RETURNS TO BANK 0
2513                                     ;*****
2513 011374 012737 000116 000114  DONE:   MOV      #PARVEC+2,@#PARVEC ;RESTORE TRAPCATCHER
2514 011402 012706 000510          MOV      #STKPT,SP    ;REINITIALIZE STACK POINTER
2515 011406 004767 002116          JSR      %7,CLRPAR    ;CLEAR ALL PARITY REGISTERS
2516 011412 005267 167126          INC      PASCNT       ;KEEP TRACK OF PASSES COMPLETED
2517 011416 004567 003512          JSR      R5,OACNV

```

```

2518 011422 000544 PASCNT
2519 011424 017106 MPCNT
2520 011426 000006 6
2521 011430 104000 TYPE ;TYPE BELL, "END PASS=" AND PASS COUNT
2522 011432 017070 MPGEND
2523 011434 013705 000042 LOGICAL: MOV 2#42,R5 ;LOADED BY MONITOR?
2524 011440 001405 BEQ CONT ;BRANCH IF NO
2525 011442 000005 RESET ;SETUP FOR MONITOR EXIT
2526 011444 004715 SENDAD: JSR 7,(5)
2527 011446 000240 NOP
2528 011450 000240 NOP
2529 011452 000240 NOP
2530 011454 032777 000400 167416 CONT: BIT #BITB,2SWR ;SWITCH B SET?
2531 011462 001401 BEQ .+4
2532 011464 000000 HALT ;HALT AT END OF PASS SET
2533 011466 105767 167420 TSTB $TPFLG
2534 011472 001006 BNE 15 ;IF NO TERMINAL, SKIP
2535 011474 105777 167404 TSTB 2TPS ;WAIT FOR ITY TO FINISH SO THAT RESET
2536 011500 100375 BPL .-4 ;WON'T CLOBBER THE BELL
2537 011502 112777 000000 167376 MOVB #0,2TPB ;OUTPUT A NULL
2538 011510 000167 170336 15: JMP BEGIN
2539
2540
2541
2542
2543
2544 ;*****
2545 ;CREATE MAP INDICATING WHERE 4K BLOCKS OF MEMORY ARE PRESENT
2546 ;*****
2546 011514 012737 011720 000004 MAPMEM: MOV #MAPMB,2#4 ;SET NO MEM MANAGEMENT TRAP
2547 011522 005737 177572 TST 2#SRO ;IS KT PRESENT? (TIMEOUT IF NO)
2548
2549 ;MAP MEMORY USING KT11 - MAX OF 124K POSSIBLE
2550 011526 005067 167020 MAPMA: CLR NOKT ;INDICATE KT11 PRESENT
2551 011532 004767 001672 JSR %7,NRALL ;INITIALLY SET ALL PAGES NONRESIDENT, BANK 0
2552 011536 004767 002036 JSR %7,MAP1 ;MAP KERNEL 0 TO BANK 0, RW
2553 ;MAP KERNEL 7 TO THE EXTERNAL BANK, RW
2554 ;MAP KERNEL 1 RW, AND TURN ON KT11
2555 011542 005067 167000 CLR TBANK
2556 011546 012767 177777 167312 MOV #177777,MEML ;SET UP CORE MAPS
2557 011554 012767 077777 167306 MOV #77777,MEMH
2558 011562 012767 000001 166730 MOV #1,BITPT ;SET UP 4K POINTER
2559 011570 012767 001066 166752 MOV #MEML,MEMUT
2560 011576 012737 011706 000004 MOV #55,2#4 ;SET UP FOR TIME OUTS
2561 011604 016777 166736 167434 25: MOV TBANK,2KPAR1 ;MAP KERNEL PAGE 1 TO BANK BEING TESTED
2562 011612 005737 021000 TST 2#21000 ;1ST K PRESENT?
2563 011616 005737 025000 TST 2#25000 ;2ND K PRESENT?
2564 011622 005737 031000 TST 2#31000 ;3RD K PRESENT?
2565 011626 005737 035000 TST 2#35000 ;4TH K PRESENT?
2566 011632 062767 000200 166706 35: ADD #200,TBANK ;UPDATE TEST ADDRESS
2567 011640 006367 166654 ASL BITPT ;UPDATE BANK POINTER
2568 011644 103006 BCC 45 ;BRANCH IF NOT DONE WITH 64K SECTION
2569 011646 012767 000001 166644 MOV #1,BITPT ;YES, DO MEMH(64-124K)
2570 011654 012767 001070 166666 MOV #MEMH,MEMUT
2571 011662 022767 007600 166656 45: CMP #7600,TBANK ;EXTERNAL BANK YET?
2572 011670 003345 BGT 25 ;NO, NOT YET
2573 011672 005037 177572 CLR 2#SRO ;YES- DISABLE KT11

```


K04

DCMFA-D, MEMORY PARITY TEST
DCMFA0.P11 25-OCT-76 22:48

MACY11 27(1006) 25-OCT-76 22:52 PAGE 48

```

2574 011676 012737 000006 000004      MOV      #6,2#4      ;RESTORE TRAPCATCHER
2575 011704 000207                RTS      %7          ;RETURN
2576 011706 046777 166606 166634 5$:  BIC      BITPT,2#MEMUT ;TIMEOUT OCCURRED-CLEAR BIT TO INDICATE
2577                ;4K BLOCK NOT PRESENT
2578 011714 022626                CMP      (SP)+,(SP)+ ;ADJUST STACK
2579 011716 000745                BR       3$         ;CHECK NEXT BLOCK
2580
2581                ;NO KT PRESENT - MAP MAX OF 28K IN 4K CONTIGUOUS BLOCKS
2582 011720 012767 000001 166624 MAPMB: MOV      #1,NOKT      ;SET FLAG TO INDICATE KT11 NOT PRESENT
2583 011726 022626                CMP      (SP)+,(SP)+ ;RESTORE STACK POINTER
2584 011730 012737 012030 000004      MOV      #3$,2#4     ;SET UP TIMEOUT RETURN
2585 011736 012767 000177 167122      MOV      #177,MEML   ;INITIALIZE MAP
2586 011744 005067 167120                CLR      MEMH
2587 011750 012767 000001 166542      MOV      #1,BITPT    ;SETUP 4K POINTER
2588 011756 005001                CLR      R1          ;INITIALIZE BANK ADDRESS
2589 011760 005761 001000 1$:      TST      1000(1)     ;1ST K PRESENT
2590 011764 005761 005000                TST      5000(1)     ;2ND K PRESENT
2591 011770 005761 011000                TST      11000(1)    ;3RD K PRESENT
2592 011774 005761 015000                TST      15000(1)    ;4TH K PRESENT
2593 012000 062701 020000 2$:      ADD      #20000,R1   ;UPDATE TEST ADDRESS
2594 012004 006367 166510                ASL      BITPT       ;UPDATE POINTER TO NEXT 4K
2595 012010 022767 000200 166502      CMP      #200,BITPT  ;28K CHECKED YET?
2596 012016 003360                BGT      1$         ;NO, CHECK NEXT 4K BLOCK
2597 012020 012737 000006 000004      MOV      #6,2#4
2598 012026 000207                RTS      %7
2599 012030 046767 166464 167030 3$:  BIC      BITPT,MEML ;TIMEOUT OCCURRED- CLEAR BIT TO
2600                ;INDICATE 4K BLOCK NOT PRESENT
2601 012036 022626                CMP      (SP)+,(SP)+ ;ADJUST STACK POINTER
2602 012040 000757                BR       2$
2603
2604
2605
2606                ;*****
2607                ;MAP PARITY CORE AND CORRESPONDENCE TO ASSOCIATED REGISTERS
2608                ;*****
2609 012042 013767 001066 166500 MAPREG: MOV      2#MEML, MEMUT ;LOAD MAP OF MEMORY PRESENT IN LOWER 64K
2610 012050 012767 000001 166442      MOV      #1,BITPT    ;INITIALIZE 4K POINTER
2611 012056 012767 000001 167136      MOV      #1,KTSTART  ;INDICATE KT11 NOT IN USE
2612 012064 005067 166476                CLR      RELOC
2613 012070 005067 166452                CLR      TBANK
2614 012074 005067 166454                CLR      HIWORD
2615                ;INITIALIZE ADDRESS OF TEST BANK
2616 012100 005067 167120                CLR      ADRTYP     ;CLEAR FLAG TO INDICATE FIRST 64K
2617                ;BEING CHECKED
2618
2619                ;SET WRITE WRONG PARITY IN ALL REGISTERS PRESENT
2620                ;THEN WRITE TEST LOCATION VIA DAT0 AND READ TEST LOCATION VIA DAT1
2621                ;THEN CLEAR WRITE WRONG PARITY IN ALL REGISTERS
2621 012104 012702 000570 MAPRB: MOV      #MPRO,R2 ;LOAD ADDRESS OF TABLE
2622 012110 032712 000001 1$:      BIT      #1(2)      ;IS THIS REGISTER PRESENT?
2623 012114 001003                BNE      .+10        ;NO, GET NEXT ONE
2624 012116 012772 000004 000000      MOV      #WWP,2(2)  ;YES, SET WRITE WRONG PARITY AND CLEAR REST
2625 012124 062702 000010                ADD      #10,R2
2626 012130 020227 000770                CMP      R2,#TREG   ;DONE WITH TABLE?
2627 012134 103765                BLO      1$         ;BRANCH IF NOT
2628 012136 016703 166404                MOV      TBANK,R3   ;LOAD ADDRESS OF 4K BANK UNDER TEST
2629 012142 066703 167056                ADD      ADRTYP,R3  ;ADD ADDRESS OFFSET (EITHER 0,2,4,OR 6)

```



```

2742 012722 012767 000001 165572 MOV #1,TRFLG ;NO- SET FLAG INDICATING TRANSITION
2743 012730 004567 002306 JSR RS,BDCNV ;CONVERT K CORE TO ASCII
2744 012734 000526 TYCOR
2745 012736 016661 MTYCOR
2746 012740 000003 3
2747 012742 104000 TYPE ;TYPE "CONTROLS", AND ADDRESS OF CORE
2748 012744 017173 MX2
2749 012746 104000 TYPE
2750 012750 016661 MTYCOR
2751 012752 104000 TYPE
2752 012754 016666 MDASH
2753 012756 005267 165542 INC TYFLG ;INDICATE TYPED
2754 012762 000422 BR TMAPD
2755 012764 005767 165532 TMAPC: TST TRFLG ;DID THIS PARITY REGISTER CONTROL PREVIOUS 4K?
2756 012770 001417 BEQ TMAPD ;NO, SKIP PRINTING
2757 012772 005067 165524 CLR TRFLG ;YES, TRANSITION OCCURRED- CLEAR FLAG
2758 012776 004567 002240 JSR RS,BDCNV ;CONVERT K CORE TO ASCII
2759 013002 000526 TYCOR
2760 013004 016661 MTYCOR
2761 013006 000003 3
2762 013010 104000 TYPE
2763 013012 016661 MTYCOR ;TYPE RIGHT AND RETURN
2764 013014 104000 TYPE
2765 013016 016674 MK
2766 013020 104000 TYPE
2767 013022 016671 MCR
2768 013024 005267 165474 INC TYFLG ;INDICATE TYPED
2769 013030 006367 165464 TMAPD: ASL BITPT ;UPDATE BIT POINTER TO NEXT 4K
2770 013034 103320 BCC TMAPB ;TEST NEXT 4K IF NOT DONE WITH 1ST 64K
2771 013036 005767 165512 TST HIWORD ;64-124K DONE?
2772 013042 001405 BEG 15 ;NO, BRANCH
2773 013044 005767 165454 TST TYFLG ;YES, WAS ANY PARITY MEMORY
2774 ;FOUND FOR THIS REGISTER?
2775
2776 013050 001250 BNE TMAPA
2777 013052 104002 ERROR ;NO PARITY MEMORY WAS FOUND FOR THIS
2778 ;REGISTER- EITHER WRITE WRONG PARITY
2779 ;FAILED, PARITY ERROR GENERATE OR
2780 ;DETECT FAILED, OR THE PARITY ERROR
2781 ;BIT FAILED TO SET
2782 013054 000646 BR TMAPA
2783 013056 016167 000004 165432 15: MOV 4(1),ADRPT ;UPDATE TO MAP WORD FOR THE UPPER 64K
2784 013064 012767 000001 165426 MOV #1,BITPT ;RESET BIT POINTER
2785 013072 005267 165456 INC HIWORD ;INDICATE LOW 64K DONE
2786 013076 000677 BR TMAPB ;LOOP
2787 013100 000207 TMAPEX: RTS ;RETURN WHEN DONE
2788
2789
2790
2791 ;*****
2792 ;ERROR HANDLER
2793 ;*****
2794 ;ERRORS CALL ENTERS HERE
2795 ;TYPES PC, ICNT, MPR ADDRESS, AND MPR CONTENTS
2796 ;TREG SHOULD CONTAIN ADDRESS OF PARITY REGISTER
2797 013102 012767 016235 000266 ERRST: MOV #MSTR,ERRB ;SETUP TO TYPE MPR ADDRESS AND CONTENTS

```

2798	013110	012767	177777	000262	MOV	#-1,ERRBX	
2799	013116	012767	000240	000256	MOV	#240,ERRBX+2	;NOP LOCATION AFTER MESSAGE
2800	013124	017767	165640	165406	MOV	@TREG,TRDATA	;SETUP DATA
2801	013132	004567	001776		JSR	R5,OACNV	;CONVERT TO ASCII
2802	013136	000770			TREG		
2803	013140	016244			MTREG		
2804	013142	000006			6		
2805	013144	004567	001764		JSR	R5,OACNV	;CONVERT TO ASCII
2806	013150	000540			TRDATA		
2807	013152	016266			MDATA		
2808	013154	000006			6		
2809	013156	000461			BR	ERRA	
2810							
2811							
2812							
2813							
2814							
2815							
2816							
2817							
2818							
2819	013160	012767	016235	000210	ERRP: MOV	#MSTR,ERRB	;IN ADDITION TO BASIC PRINTOUT, TYPE
2820							;MPR ADDRESS AND CONTENTS
2821	013166	012767	016277	000204	MOV	#MSTRX,ERRBX	;ALSO OUTPUT DATA EXPECTED AND ACTUAL
2822	013174	012767	177777	000200	MOV	#-1,ERRBX+2	;NOP LOCATION AFTER MESSAGE
2823	013202	010567	165324		MOV	R5,TSTLOC	;STORE ADDRESS BEING TESTED
2824	013206	017767	165556	165324	MOV	@TREG,TRDATA	
2825	013214	004567	001714		JSR	R5,OACNV	
2826	013220	000770			TREG		
2827	013222	016244			MTREG		
2828	013224	000006			6		
2829	013226	004567	001702		JSR	R5,OACNV	
2830	013232	000540			TRDATA		
2831	013234	016266			MDATA		
2832	013236	000006			6		
2833	013240	004567	001670		JSR	R5,OACNV	
2834	013244	000532			TSTLOC		
2835	013246	016323			MSTRX1		
2836	013250	000006			6		
2837	013252	004567	001656		JSR	R5,OACNV	
2838	013256	000534			SHOBE		
2839	013260	016337			MSTRX3		
2840	013262	000006			6		
2841	013264	004567	001644		JSR	R5,OACNV	
2842	013270	000536			WAS		
2843	013272	016353			MSTRX5		
2844	013274	000006			6		
2845	013276	000411			BR	ERRA	
2846							
2847							
2848							
2849							
2850							
2851	013300	012767	177777	000070	ERR: MOV	#-1,ERRB	;SET UP ONE MESSAGE CALL
2852	013306	012767	000240	000064	MOV	#240,ERRBX	
2853	013314	012767	000240	000060	MOV	#240,ERRBX+2	

```

;ERRORP CALL ENTERS HERE
;TYPE PC ICNT MPR ADDRESS MPR CONTENTS
;TEST LOCATION ADDRESS VALUE EXPECTED VALUE FOUND
;R5 MUST CONTAIN ADDRESS OF TEST LOCATION
;SHOBE MUST CONTAIN EXPECTED VALUE
;WAS MUST CONTAIN ACTUAL DATA
;TREG MUST CONTAIN ADDRESS OF PARITY REGISTER
    
```

```

2854 013322 032777 020000 165550  ERRA:  BIT      #BIT13, @SWR      ;INHIBIT ERROR PRINT?
2855 013330 001025          BNE      ERRC      ;YES- BRANCH
2856 013332 011667 000070          MOV      (SP), ERRO ;NO- DEVELOP CALLING ADDRESS
2857 013336 162767 000002 000062  SUB      #2, ERRO
2858 013344 004567 001564          JSR      R5, OACNV  ;GO TO OCTAL TO ASCII CONVERT
2859 013350 013426          ERRA1:  ERRO      ;SOURCE ADDRESS
2860 013352 016210          ERRA2:  MPC      ;DESTINATION ADDRESS
2861 013354 000006          B        6         ;#OF DIGITS TO CONVERT
2862 013356 004567 001552          JSR      R5, OACNV  ;CONVERT ICNT TO ASCII
2863 013362 016176          ERRA3:  ICNT
2864 013364 016226          ERRA4:  MICNT
2865 013366 000006          B        6
2866 013370 004567 001506          JSR      R5, TYPX
2867 013374 016202          ERRA5:  MEQ
2868 013376 000000          ERRA6:  OPEN      ;TYPE MESSAGE
2869 013400 000000          ERRA7:  OPEN      ;ERROR HEADER
2870 013402 177777          -1
2871 013404 023737 000042 000046  ERRC:   CMP      @#42, @#46 ;ARE WE IN ACT11 AUTOMATIC MODE?
2872 013412 001403          BEQ      .+10      ;YES, HALT ON ERROR
2873 013414 005777 165460          TST      @SWR      ;HALT ON ERROR SET?
2874 013420 100001          BPL      .+4
2875 013422 000000          HALT
2876 013424 000002          RTI
2877 013426 000000          ERRO:   OPEN
2878
2879
2880
2881          ;MAP ALL PAGES NON-RESIDENT, BANK 0
2882 013430 013746 000004  NRALL:  MOV      @#4, -(SP)
2883 013434 013746 000006          MOV      @#6, -(SP)
2884 013440 012737 000006 000004          MOV      @#6, @#4
2885 013446 012737 000002 000006          MOV      @RTI, @#6
2886 013454 010146          MOV      R1, -(SP)
2887 013456 010246          MOV      R2, -(SP)
2888 013460 010346          MOV      R3, -(SP)
2889 013462 012701 001226          MOV      @PORTAB, R1
2890 013466 012703 000040 1$:     MOV      @32, R3
2891 013472 012102          MOV      (R1)+, R2
2892 013474 005022 2$:     CLR      (R2)+
2893 013476 005303          DEC      R3
2894 013500 001375          BNE      2$
2895 013502 020127 001232          CMP      R1, @PDREND
2896 013506 003767          BLE      1$
2897 013510 012603          MOV      (SP)+, R3
2898 013512 012602          MOV      (SP)+, R2
2899 013514 012601          MOV      (SP)+, R1
2900 013516 012637 000006          MOV      (SP)+, @#6
2901 013522 012637 000004          MOV      (SP)+, @#4
2902 013526 000207          RTS      %7
2903
2904
2905
2906          ;ROUTINE TO CLEAR ALL PARITY REGISTERS PRESENT
2907 013530 010146  CLRPAR: MOV      R1, -(SP)
2908 013532 010246          MOV      R2, -(SP)
2909 013534 010701          MOV      PC, R1

```

```

2910 013536 062701 165032
2911 013542 010702
2912 013544 062702 165224
2913 013550 032711 000001
2914 013554 001002
2915 013556 005071 000000
2916 013562 062701 000010
2917 013566 020102
2918 013570 103767
2919 013572 012602
2920 013574 012601
2921 013576 000207
2922
2923
2924
2925
2926
2927
2928 013600 012777 077406 165426
2929 013606 012777 077406 165422
2930 013614 012777 077406 165420
2931 013622 012777 007600 165422
2932 013630 005077 165410
2933 013634 012737 000001 177572
2934 013642 000207
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944 013644 010246
2945 013646 012702 000200
2946 013652 012701 000002
2947 013656 005767 164670
2948 013662 001403
2949 013664 022701 000200
2950 013670 101420
2951
2952 013672 030160 000002
2953 013676 001021
2954 013700 062702 000200
2955 013704 006301
2956 013706 103363
2957 013710 012701 000001
2958 013714 030160 000004
2959 013720 001010
2960 013722 062702 000200
2961 013726 006301
2962 013730 103371
2963 013732 012701 000001
2964 013736 012602
2965 013740 000207
    
```

```

ADD #MPRO-. ,R1
MOV PC,R2
ADD #TREG-. ,R2
1$: BIT #1,R1 ;IS THIS REGISTER PRESENT?
BNE .+6
CLR @R1 ;CLEAR ALL PARITY REGISTERS
ADD #10,R1
CMP R1,R2
BLO 1$
MOV (SP)+,R2
MOV (SP)+,R1
RTS %7

;ROUTINE TO MAP KERNEL 0 TO BANK 0 READ/WRITE.
;KERNEL 1 READ/WRITE BUT BANK MAPPED BY CALLING ROUTINE.
;AND KERNEL 7 TO EXTERNAL BANK, READ/WRITE
MAP1: MOV #77406,@KPRO
MOV #77406,@KPDR1
MOV #77406,@KPDR7
MOV #7600,@KPAR7
CLR @KPAR0
MOV #1,@#SRO
RTS %7

;ROUTINE TO LOCATE THE FIRST PARITY MEMORY ADDRESS (ABOVE BANK 0)
;CORRESPONDING TO A GIVEN PARITY REGISTER-REQUIRES THAT THE ROUTINES
;MAPMEM AND MAPREG HAVE ALREADY BEEN RUN
;TO USE, PUT THE ADDRESS OF THE REGISTER IN R0 (I.E. POINT TO
;TO MAP TABLE)-THE DESIRED ADDRESS IS RETURNED IN R1, USING KERNEL PAGE 1 IF
;KT11 IS PRESENT
LOCATM: MOV R2,-(SP)
MOV #200,R2
MOV #2,R1 ;SKIP USE OF BANK 0
1$: TST NOKT ;KT11 PRESENT?
BEQ 3$ ;YES, BRANCH
CMP #200,R1 ;NO, CHECK ONLY FOR MEMORY IN FIRST 28K
BLOS 4$ ;IF NO MEMORY FOUND IN 1ST 28K, GIVE
;ERROR RETURN
3$: BIT R1,2(R0) ;DOES THIS 4K CORRESPOND TO THIS REGISTER?
BNE LOCAT1 ;YES, BRANCH
ADD #200,R2 ;NO, CHECK TO SEE IF NEXT 4K CORRESPONDS
ASL R1
BCC 1$
MOV #1,R1
2$: BIT R1,4(R0) ;CHECK HIGH 64K
BNE LOCAT1
ADD #200,R2
ASL R1
BCC 2$
4$: MOV #1,R1 ;NO PARITY MEMORY CORRESPONDS TO
MOV (SP)+,R2 ;THIS REGISTER- RETURN WITH ERROR
RTS %7 ;INDICATION
    
```

```

2966 013742 005767 164604 LOCAT1: TST NOKT ;KT11 PRESENT?
2967 013746 001005 BNE 1$ ;NO- BRANCH OVER
2968 013750 010277 165272 MOV R2,JKPAR1 ;YES- SETUP R1 TO REFERENCE 4K
2969 013754 012701 020000 MOV #20000,R1 ;BANK USING KERNEL PAGE 1
2970 013760 000407 BR 2$
2971 013762 010201 1$: MOV R2,R1 ;SETUP R1 TO REFERENCE 4K BANK
2972 013764 006301 ASL R1 ;WITHOUT KT11
2973 013766 006301 ASL R1
2974 013770 006301 ASL R1
2975 013772 006301 ASL R1
2976 013774 006301 ASL R1
2977 013776 006301 ASL R1
2978 014000 116067 000006 000010 2$: MOV#B 6(R0),LSAV
2979 014006 066701 000004 ADD LSAV,R1
2980 014012 012602 MOV (SP)+,R2 ;RESTORE R2
2981 014014 000207 RTS %7 ;AND RETURN
2982 014016 000000 LSAV: 0
2983
2984
2985
2986 ;SAVE LOADER IN TOP OF FIRST 4K
2987 014020 013746 000004 SAVLDR: MOV #4,-(SP) ;SAVE CONTENTS OF TIMEOUT VECTOR
2988 014024 013746 000006 MOV #6,-(SP)
2989 014030 012737 000006 000004 MOV #6,#4
2990 014036 012737 000002 000006 MOV #RTI,#6 ;SETUP TO RTI ON TIMEOUT
2991 014044 010146 MOV R1,-(SP) ;SAVE REGISTERS
2992 014046 010246 MOV R2,-(SP)
2993 014050 012701 157500 MOV #157500,R1
2994 014054 000261 1$: SEC
2995 014056 005711 TST #R1 ;IF TIMEOUT, C BIT WILL STILL BE SET
2996 014060 103006 BCC 2$ ;IF NO TIMEOUT, C BIT WILL BE CLEAR
2997 014062 102701 020000 SUB #20000,R1 ;TIMEOUT OCCURRED, CHECK FOR NEXT LOWER BANK
2998 014066 020127 020000 CMP R1,#20000
2999 014072 101370 BHI 1$
3000 014074 000410 BR SAVLDR
3001 014076 012702 017500 2$: MOV #17500,R2 ;ONLY 4K OF 1ST 28K PRESENT- EXIT
3002 014102 012122 3$: MOV (R1)+(R2)+ ;THIS BANK IS HIGHEST OF 1ST 28K- COPY
3003 014104 020227 020000 CMP R2,#20000 ;LOADER AREA TO BANK 0
3004 014110 103774 BLO 3$
3005 014112 005267 000016 INC LDRSVD ;INDICATE LOADER HAS BEEN MOVED TO
3006 ;THE TOP OF THE FIRST 4K
3007 014116 012602 SAVLDR: MOV (SP)+,R2
3008 014120 012601 MOV (SP)+,R1
3009 014122 012637 000006 MOV (SP)+,#6
3010 014126 012637 000004 MOV (SP)+,#4
3011 014132 000207 RTS %7
3012 014134 000000 LDRSVD: 0
3013
3014
3015
3016 ;ROUTINE TO RESTORE THE LOADER FROM BANK 0 (WHERE IT WAS SAVED) TO THE
3017 ;HIGHEST BANK IN THE FIRST 28K OF MEMORY
3018 014136 005767 177772 RSTLDR: TST LDRSVD
3019 014142 001431 BEQ RSTLDR
3020 014144 012737 000006 000004 MOV #6,#4
3021 014152 012737 000002 000006 MOV #RTI,#6

```



```

3022 014160 012701 157500
3023 014164 000261
3024 014166 005711
3025 014170 103006
3026 014172 162701 020000
3027 014176 020127 020000
3028 014202 101370
3029 014204 000410
3030 014206 012702 017500
3031 014212 012221
3032 014214 020227 020000
3033 014220 103774
3034 014222 104000
3035 014224 017205
3036 014226 000000
3037 014230 000776
3038
3039
3040
3041
3042
3043 014232 010146
3044 014234 010246
3045 014236 010346
3046 014240 010446
3047 014242 013746 000004
3048 014246 013746 000006
3049 014252 013746 000114
3050 014256 013746 000116
3051 014262 012737 000006 000004
3052 014270 005037 000006
3053 014274 012737 000116 000114
3054 014302 005037 000116
3055 014306 005767 164240
3056 014312 001513
3057 014314 005002
3058 014316 012703 000001
3059 014322 004767 177202
3060 014326 005712
3061 014330 000240
3062 014332 012701 000570
3063
3064 014336 032711 000001
3065 014342 001003
3066 014344 005771 000000
3067 014350 100424
3068 014352 062701 000010
3069 014356 020127 000770
3070 014362 103765
3071 014364 062702 000002
3072 014370 032702 017777
3073 014374 001352
3074 014376 006303
3075 014400 020327 000200
3076 014404 103035
3077 014406 030367 164454
    
```

```

1S:  MOV #157500,R1
      SEC
      TST @R1 ; IF TIMEOUT, C BIT WILL BE SET
      BCC 2S ; IF NO TIMEOUT, C BIT WILL BE CLEAR
      SUB #20000,R1 ; TIMEOUT OCCURRED, CHECK FOR NEXT
      CMP R1,#20000 ; LOWER BANK
      BHI 1S
      BR RSTLDX
2S:  MOV #17500,R2
3S:  MOV (R2)+,(R1)+
      CMP R2,#20000
      BLO 3S
      TYPE ; TYPE MESSAGE "LOADER RESTORED"
      LDRMSG
RSTLDX: HALT ; LOADER HAS BEEN RESTORED
        BR -.2 ; TO HIGHEST BANK IN FIRST 28K

; SCAN ALL MEMORY FOR BAD PARITY, TYPE 18 BIT ADDRESSES OF
; LOCATIONS FOUND TO BE BAD, AND WRITE INTO LOCATIONS WITH GOOD PARITY
PSCAN: MOV R1,-(SP) ; STORE REGISTERS AND LOCATIONS TO BE
        MOV R2,-(SP) ; ALTERED
        MOV R3,-(SP)
        MOV R4,-(SP)
        MOV @#4,-(SP)
        MOV @#6,-(SP)
        MOV @#14,-(SP)
        MOV @#16,-(SP)
        MOV @#6,@#4 ; SETUP TIMEOUT TRAPCATCHER
        CLR @#6
        MOV @#16,@#14 ; SETUP PARITY TRAP TRAPCATCHER
        CLR @#16
        TST NOKT ; KTI1 PRESENT?
        BEQ PSCAN1 ; YES, BRANCH
        CLR R2 ; R2 CONTAINS TEST ADDRESS
        MOV #1,R3 ; R3 USED AS A BIT POINTER
1S:  JSR %7,CLRPAR ; CLEAR ALL PARITY REGISTERS
      TST @R2 ; READ LOCATION TO CHECK FOR BAD PARITY
      NOP
      MOV #MPRO,R1 ; SETUP TO SCAN REGISTERS FOR PARITY
      ; ERROR SET
3S:  BIT #1,@R1
      BNE .+10
      TST @R1 ; PARITY ERROR SET?
      BMI 6S ; YES- BRANCH
      ADD #10,R1 ; NO- CHECK NEXT REGISTER
      CMP R1,#TREG
      BLO 3S ; LOOP UNTIL ALL REGISTERS HAVE BEEN CHECKED
4S:  ADD #2,R2 ; MOVE ADDRESS POINTER
      BIT #17777,R2 ; DONE WITH 4K?
      BNE 1S ; NO, CONTINUE
5S:  ASL R3 ; YES, CHECK FOR TESTING NEXT 4K
      CMP R3,#200
      BHIS PSCANX ; EXIT IF DONE WITH 28K
      BIT R3,MEML ; IS THIS MEMORY PRESENT?
    
```


3134	014634	006303		ASL	R3	
3135	014636	103361		BCC	PSLUP	; BRANCH IF NOT END OF 64K
3136	014640	005767	177674	TST	PSCANH	; END OF TOP 64K?
3137	014644	001003		BNE	PSCX1	; YES, GET READY TO EXIT
3138	014646	005267	177666	INC	PSCANH	; NO, SET FLAG INDICATING DONE WITH
3139						; LOWER 64K
3140	014652	000751		BR	PSLOOP	
3141	014654	012677	164366	PSCX1: MOV	(SP)+, @KPAR1	
3142	014660	000707		BR	PSCANX	
3143	014662	012702	020000	PSXTST: MOV	#20000, R2	; R2 USED AS ADDRESS POINTER
3144	014666	004767	176636	1S: JSR	%7, CLPAR	; CLEAR ALL PARITY REGISTERS
3145	014672	005712		TST	@R2	; READ LOCATION
3146	014674	012701	000570	MOV	#MPRO, R1	; SETUP TO SCAN REGISTERS FOR PARITY ERROR SET
3147	014700	032711	000001	2S: BIT	#1, @R1	
3148	014704	001003		BNE	.+10	
3149	014706	005771	000000	TST	@(R1)	; PARITY ERROR SET?
3150	014712	100413		BMI	4S	; YES, BRANCH
3151	014714	062701	000010	ADD	#10, R1	; NO, CHECK NEXT
3152	014720	020127	000770	CMP	R1, @TREG	
3153	014724	103765		BLO	2S	; LOOP UNTIL ALL REGISTERS HAVE BEEN CHECKED
3154	014726	062702	000002	3S: ADD	#2, R2	; UPDATE TEST ADDRESS POINTER
3155	014732	020227	040000	CMP	R2, #40000	; DONE WITH BANK?
3156	014736	103753		BLO	1S	; NO- LOOP
3157	014740	000732		BR	PSNXT	; YES- GO CHECK FOR ANOTHER BANK
3158	014742	010267	177570	4S: MOV	R2, PSADRS	; PARITY ERROR OCCURRED- GET 18 BIT
3159	014746	042767	160000	BIC	#160000, PSADRS	; OCTAL ADDRESS OF BAD LOCATION
3160	014754	005046		CLR	-(SP)	
3161	014756	017746	164264	MOV	@KPAR1, -(SP)	
3162	014762	006316		ASL	@SP	
3163	014764	006316		ASL	@SP	
3164	014766	006316		ASL	@SP	
3165	014770	006316		ASL	@SP	
3166	014772	006316		ASL	@SP	
3167	014774	006166	000002	ROL	2(SP)	
3168	015000	006316		ASL	@SP	
3169	015002	006166	000002	ROL	2(SP)	
3170	015006	006366	000002	ASL	2(SP)	
3171	015012	062667	177520	ADD	(SP)+, PSADRS	
3172	015016	004567	000112	JSR	R5, OACNV	; CONVERT LOW 16 OCTAL BITS TO ASCII
3173	015022	014536		PSADRS		
3174	015024	017145		MPSER1		
3175	015026	000006		6		
3176	015030	116704	002111	MOVB	MPSER1, R4	
3177	015034	062604		ADD	(SP)+, R4	; CHANGE TO ASCII FOR 18 BITS
3178	015036	110467	002103	MOVB	R4, MPSE1	
3179	015042	104000		TYPE		; TYPE ADDRESS OF LOCATION WITH BAD PARITY
3180	015044	017115		MPSE1		
3181	015046	032777	002000	BIT	#2000, @SWR	; SWITCH 10 SET?
3182	015054	001401		BEQ	.+4	; NO- BRANCH
3183	015056	000000		HALT		; HALT ON BAD PARITY SET
3184	015060	011212		MOV	@R2, @R2	; REWRITE LOCATION CONTAINING BAD PARITY
3185	015062	005071	000000	CLR	@(R1)	; CLEAR PARITY ERROR BIT
3186	015066	005712		TST	@R2	; READ LOCATION TO SEE IF PARITY IS NOW GOOD
3187	015070	005771	000000	TST	@(R1)	; CHECK PARITY ERROR BIT
3188	015074	100001		BPL	.+4	
3189	015076	104002		ERROR		; REWRITING LOCATION DID NOT CLEAR BAD PARITY

```

3190 015100 000712          BR      3$          :GO TEST NEXT LOCATION
3191
3192
3193
3194
3195
3196 015102 012567 000022          :PIC ROUTINE TO OUTPUT A SERIES OF ASCII MESSAGES (CALLED VIA JSR R5)
3197 015106 022767 177777 000014 TYP$X: MOV      (R5)+,TYP$BX          ;GET ADDRESS OF MESSAGE
3198 015114 001001          CMP      #-1,TYP$BX          ;TERMINATOR?
3199 015116 000205          BNE     TYP$AX          ;NO BRANCH
3200 015120 013746 177776 TYP$AX: MOV      @PS,-(SP)          ;YES RETURN
3201 015124 004767 163764          JSR     PC,$TYPE          ;SETUP TO CALL TYPE ROUTINE VIA JSR
3202 015130 000000          TYP$BX: OPEN     PC,$TYPE          ;TYPE ASCII MESSAGE
3203 015132 000763          BR      TYP$X
3204
3205
3206
3207          :SUBROUTINE FOR OCTAL TO ASCII CONVERSION
3208 015134 013567 000074 OACNV: MOV      @($)+,OACNVX          ;GET OCTAL VALUE
3209 015140 012567 000072          MOV      ($)+,OACDST          ;GET DESTINATION ADDRESS
3210 015144 012567 000070          MOV      ($)+,OACNT          ;GET CONVERT COUNT
3211 015150 066767 000064 000060 OACNVA: ADD     OACNT,OACDST          ;DEVELOP ADDRESS TO STORE 1ST CHAR.
3212 015156 016746 000052          MOV      OACNVX,-(SP)
3213 015162 042716 177770          BIC     #177770,@SP          ;ISOLATE LEAST SIGNIFICANT DIGIT
3214 015166 062716 000060          ADD     #60,@SP          ;CONVERT DIGIT TO ASCII
3215 015172 005367 000040          DEC     OACDST
3216 015176 112677 000034 000024 MOVB    (SP)+,@OACDST          ;STORE ASCII CHARACTER
3217 015202 042767 000007          BIC     #7,OACNVX
3218 015210 006067 000020          ROR     OACNVX
3219 015214 006067 000014          ROR     OACNVX
3220 015220 006067 000010          ROR     OACNVX
3221 015224 005367 000010          DEC     OACNT
3222 015230 001352          BNE     OACNVA          ;DONE ALL DIGITS?
3223 015232 000205          RTS     RS          ;BRANCH IF NOT DONE
3224 015234 000000          :DONE, EXIT
3225 015236 000000 OACNVX: OPEN
3226 015240 000000 OACDST: 0
3227          OACNT: 0
3228
3229
3230          :SUBROUTINE FOR BINARY TO DECIMAL ASCII CONVERSION
3231 015242 104005 BDCNV: SAVD4          ;SAVE REGS
3232 015244 012700 015420          MOV     #DECVAL,%0          ;SET UP ADDR TO STORE DECIMAL ASCII
3233 015250 013501          MOV     @($)+,R1          ;BINARY VALUE TO R1
3234 015252 012567 000052          MOV     ($)+,BDCNVC          ;DESTINATION ADDR TO BDCNVC
3235 015256 012567 000050          MOV     ($)+,BDCNVD          ;CHARACTER COUNT TO BDCNVD
3236 015262 012702 015406          MOV     #ADTEMP,R2          ;ADDR OF TEN POWER STRING
3237 015266 012767 000005 000104 BDCNVA: MOV     #5,CNVCTR          ;SET UP FOR 5 POWER CONVERSIONS
3238 015274 012267 000104          MOV     (2)+,TENPWR          ;MOVE POWER OF TEN VALUE
3239 015300 004767 000034          JSR     PC,SUBTEN          ;PERFORM CONVERSION
3240 015304 005367 000070          DEC     CNVCTR          ;DONE 5 CONVERSIONS?
3241 015310 001371          BNE     BDCNVA          ;BRANCH IF NOT YET 5.
3242 015312 166700 000014          SUB     BDCNVD,%0
3243 015316 010067 000004          MOV     %0,BDCNVB
3244 015322 004567 000100          JSR     RS,BMOVE
3245 015326 000000 BDCNVB: OPEN

```

```

3246 015330 000000          BDCNVC: OPEN
3247 015332 000000          BDCNVD: OPEN
3248 015334 104006          RST04          ;RESTORE REGS AND EXIT
3249 015336 000205          RTS          RS
3250 015340 005067 000036    SUBTEN: CLR    DIGIT
3251 015344 166701 000034    SUBTNA: SUB    TENPWR,R1          ;SUBTRACT TEN POWER FROM BINARY VALUE
3252 015350 103403          BCS          SUBTNB          ;BRANCH IF UNSUCCESSFUL SUBTRACTION
3253 015352 005267 000024    INC          DIGIT
3254 015356 000772          BR          SUBTNA
3255 015360 066701 000020    SUBTNB: ADD    TENPWR,R1          ;RESTORE SUBTRACTED VALUE.
3256 015364 062767 000060 000010    ADD          #60,DIGIT          ;CONVERT (DIGIT) TO ASCII
3257 015372 116720 000004          MOVB         DIGIT,(0)+          ;MOVE ASCII CHAR TO DECVAL FIELD
3258 015376 000207          RTS          PC          ;EXIT
3259 015400 000000          CNVCTR: OPEN
3260 015402 000000          DIGIT: OPEN
3261 015404 000000          TENPWR: OPEN
3262 015406 023420          ADTEMP: 10000.
3263 015410 001750          1000.
3264 015412 000144          100.
3265 015414 000012          10.
3266 015416 000001          1.
3267 015420          040          040          DECVAL: .BYTE 040,040,040,040,040,040
3268 015423          040          040
3269
3270
3271
3272          ;SUBROUTINE TO MOVE A VARIABLE NUMBER OF BYTES
3273 015426 104005          BMOVE: SAV04          ;SAVE REGS
3274 015430 012501          MOV          (5)+,R1          ;GET FROM ADDRESS
3275 015432 012502          MOV          (5)+,R2          ;GET TO ADDRESS
3276 015434 012503          MOV          (5)+,R3          ;GET COUNT
3277 015436 112122          BMOVA: MOVB    (1)+,(2)+          ;MOVE BYTE
3278 015440 005303          DEC          R3          ;DECREMENT COUNT
3279 015442 001375          BNE          BMOVA          ;BRANCH IF NOT DONE
3280 015444 104006          RST04          ;RESTORE REGS AND EXIT
3281 015446 000205          RTS          RS
3282
3283
3284
3285          ;UNEXPECTED POWER FAIL SERVICE
3286          ;BECAUSE WWP MAY BE SET IN MPR'S AND ALL PROCESSOR REGISTERS
3287          ;MAY BE IN USE, CONTINUATION AFTER POWER FAIL IS NOT ATTEMPTED.
3288          ;INSTEAD, THE PROGRAM RESTARTS AFTER A POWER FAILURE
3289 015450 012737 015514 000024    PWRDN: MOV     #PWRUP,#24          ;SET UP FOR POWER UP
3290 015456 012701 000570          MOV     #MPRO,R1
3291 015462 032711 000001          IS:    BIT     #1,R1
3292 015466 001002          BNE     .+6
3293 015470 005071 000000          CLR     @R1          ;CLEAR PARITY REGISTERS IN CASE
3294 015474 062701 000010          ADD     #10,R1          ;WWP IS SET
3295 015500 020127 000770          CMP     R1,#TREG

```

```

3296 015504 103766      BLO      1$
3297 015506 010667 163542  MOV      SP,SPSAV
3298 015512 000000      HALT
3299 015514 012737 015450 000024 PWRUP:  MOV      #PWRDN, @#24      ;POWER DOWN HALT
3300 015522 016706 163526      MOV      SPSAV, SP      ;SET UP FOR POWER DOWN
3301 015526 005027 000000      CLR      #0      ;STALL SO OUTPUT WON'T BE GARBLED
3302 015532 005367 177772      DEC      #-2
3303 015536 001375      BNE      #-4
3304 015540 104000      TYPE
3305 015542 017051      MPWRF
3306 015544 000167 163522      JMP      RSTART      ;RESTART
3307
3308
3309
3310      ;EMT HANDLER
3311 015550 011646      EMTINT: MOV      (SP), -(SP)      ;GET SAVED PC
3312 015552 162716 000002      SUB      #2, (SP)      ;DECREMENT PC BY 2
3313 015556 017616 000000      MOV      @ (SP), (SP)      ;GET CALL
3314 015562 121667 000050      CMPB    (SP), EMTLIM      ;CHECK IF CALL WITHIN LIMITS
3315 015566 101402      BLOS    EMTA
3316 015570 000000      HALT      ;CALL IS NOT WITHIN LIMITS
3317 015572 000776      BR      #-2
3318 015574 006116      EMTA:  ROL      (SP)      ;EMT ARG X 2
3319 015576 042716 177001      BIC      #177001, (SP)      ;REMOVE 7 MSB
3320 015602 062716 015614      ADD      #EMTTAB, (SP)      ;FORM EMT RTN ADDRESS
3321 015606 017616 000000      MOV      @ (SP), (SP)
3322 015612 000136      JMP      @ (SP)+      ;GO TO EMT RETURN
3323
3324
3325      ;EMT DEFINITIONS AND ASSIGNMENTS
3326 015614      EMTTAB:
3327      TYPE=EMT+EMTX
3328 015614 001114      $TYPE
3329      SCOPE=EMT+EMTX
3330 015616 016074      SCOPEC
3331      ERROR=EMT+EMTX
3332 015620 013300      ERR
3333      ERRORP=EMT+EMTX
3334 015622 013160      ERRP
3335      ERRORS=EMT+EMTX
3336 015624 013102      ERRST
3337      SAVO4=EMT+EMTX
3338 015626 015640      SVO4
3339      RSTO4=EMT+EMTX
3340 015630 015726      RSO4
3341      RSTO5=EMT+EMTX
3342 015632 015754      RSO5
3343      SAVO5=EMT+EMTX
3344 015634 015660      SVO5
3345 015636 000010      EMTLIM: EMTX-1
3346
3347
3348      ;SUBROUTINE TO SAVE REGS 0-4
3349 015640 012666 177764      SVO4:  MOV      (SP)+, -12.(SP)      ;MOVE PC+PS UP STACK
3350 015644 012666 177764      MOV      (SP)+, -12.(SP)
3351 015650 012767 000002 000040      MOV      #RTI, SVO5C

```

```

3352 015656 000411          BR      SV05B
3353
3354
3355
3356
3357 015660 012767 000240 000030 ;SUBROUTINE TO SAVE REGS 0-5 + PLACE EMT PC IN R5
3358 015666 000400          SV05S: MOV      #NOP,SV05C
3359          BR      SV05A
3360          ;SUBROUTINE TO SAVE REGS 0-5
3361 015670 012666 177762          SV05A: MOV      (SP)+,-14.(SP)
3362 015674 012666 177762          MOV      (SP)+,-14.(SP)
3363 015700 010546          MOV      R5,-(SP)
3364 015702 010446          SV05B: MOV      R4,-(SP)
3365 015704 010346          MOV      R3,-(SP)
3366 015706 010246          MOV      R2,-(SP)
3367 015710 010146          MOV      R1,-(SP)
3368 015712 010046          MOV      %0,-(SP)
3369 015714 024646          CMP      -(SP),-(SP)
3370 015716 000002          SV05C: RTI
3371 015720 016605 000020          MOV      16.(SP),R5          ;RTI OR NOP
3372 015724 000002          RTI          ;EMT PC TO R5
3373
3374
3375
3376          ;SUBROUTINE TO RESTORE REGS 0-4
3377 015726 022626          RS04: CMP      (SP)+,(SP)+
3378 015730 012600          MOV      (SP)+,%0
3379 015732 012601          MOV      (SP)+,R1
3380 015734 012602          MOV      (SP)+,R2
3381 015736 012603          MOV      (SP)+,R3
3382 015740 012604          MOV      (SP)+,R4
3383 015742 016646 177764          MOV      -12.(SP),-(SP)          ;MOVE PC+PS DOWN STACK
3384 015746 016646 177764          MOV      -12.(SP),-(SP)
3385 015752 000002          RTI
3386
3387
3388
3389          ;SUBROUTINE TO RESTORE REGS 0-5
3390 015754 010566 000020          RS05S: MOV      R5,16.(SP)          ;SET EMT PC TO R5
3391 015760 022626          CMP      (SP)+,(SP)+
3392 015762 012600          MOV      (SP)+,%0
3393 015764 012601          MOV      (SP)+,R1
3394 015766 012602          MOV      (SP)+,R2
3395 015770 012603          MOV      (SP)+,R3
3396 015772 012604          MOV      (SP)+,R4
3397 015774 012605          MOV      (SP)+,R5
3398 015776 016646 177762          MOV      -14.(SP),-(SP)
3399 016002 016646 177762          MOV      -14.(SP),-(SP)
3400 016006 000002          RTI
3401
3402
3403
3404          ;ROUTINE TO LOOP THRU A SINGLE INSTRUCTION TEST
3405          ;LOAD THE STARTING ADDRESS OF THE TEST
3406          ;YOU WISH TO RUN (THE ADDRESS OF THE TESTXX
3407          ;TAG) AT THE 1ST HALT, SET SWITCH REGISTER

```

M05

DCMFA-D MEMORY PARITY TEST
DCMFA-D.P11 25-OCT-76 22:48

MACY11 27(1006) 25-OCT-76 22:52 PAGE 63

```

3408
3409
3410 016010 005037 177776
3411 016014 000000
3412 016016 017767 163056 000154
3413 016024 062767 000002 000146
3414 016032 000000
3415 016034 012767 177777 162446
3416 016042 032777 010000 163030
3417 016050 001404
3418 016052 042737 000020 177776
3419 016060 000403
3420 016062 052737 000020 177776
3421 016070 000177 000104
  
```

; OPTIONS AT THE 2ND HALT.
; NOTE THAT SW11 MUST BE DOWN AFTER THE 2ND HALT

```

TESTX: CLR @#PS
        HALT
        MOV @SWR, RETURN
        ADD @2, RETURN
        HALT
        MOV @-1, TSTX
        BIT @10000, @SWR
        BEQ .+12
        BIC @20, @#PS
        BR .+10
        BIS @20, @#PS
        JMP @RETURN
  
```

; WAIT FOR STARTING ADDRESS
; LOAD STARTING ADDRESS IN RETURN
; ADD 2 TO POINT TO INSTRUCTION AFTER
; SET SR OPTIONS
; SET FLAG
; CHECK SW12
; BRANCH IF NOT SET
; CLEAR TRACE BIT
; SKIP NEXT INSTRUCTION
; SET TRACE BIT
; JUMP TO TEST

```

3422
3423
3424
3425
3426
3427 016074 032777 040000 162776
3428 016102 001020
3429 016104 032777 004000 162766
3430 016112 001021
3431 016114 005767 162424
3432 016120 001416
3433 016122 005767 162362
3434 016126 001006
3435 016130 026767 000042 000036
3436 016136 100007
3437 016140 005267 000032
3438 016144 022606
3439 016146 012677 161624
3440 016152 000177 000022
3441 016156 005067 162326
3442 016162 005067 000010
3443 016166 011667 000006
3444 016172 000002
3445 016174 000100
3446 016176 000000
3447 016200 000000
  
```

; SCOPE AND/OR ITERATION LOOP FOR EACH TEST 64 TIMES
; A SETUP ROUTINE SHOULD INITIALIZE RETURN AND IMAX

```

SCOPEC: BIT @40000, @SWR
        BNE SCOPEB
        BIT @4000, @SWR
        BNE SCOPEC
        TST PASCNT
        BEQ SCOPEG
        TST TSTX
        BNE SCOPEB
        CMP ICNT, IMAX
        BPL SCOPEG
        INC ICNT
SCOPEB: CMP (6)+, %6
        MOV (6)+, @PS
        JMP @RETURN
SCOPEG: CLR TSTX
        CLR ICNT
        MOV @%6, RETURN
  
```

; TEST SR FOR SCOPE
; YES, SCOPE
; NO-TEST FOR ITERATION
; INHIBIT ITERATION
; FIRST PASS?
; YES, INHIBIT ITERATIONS
; USING SINGLE SUBTEST STARTUP?
; YES, LOOP
; COMPARE CURRENT COUNT TO MAX NUMBER
; EXIT-DONE
; INCREMENT COUNT
; REPOSITION STACK
; RESTORE PREVIOUS PROCESSOR STATUS
; REPEAT TEST
; IF USING TESTX STARTUP, RETURN TO NORMAL FLOW
; CLEAR COUNT
; SAVE SCOPE RETURN POINTER
; RETURN INLINE-NEXT TEST
; ITERATION COUNT
; COUNT 'LOCATION FOR ITERATION LOOP
; ADDR . OF LAST TEST

```

3448
3449
3450
3451
3452
3453 016202
3454 016202 005015 041520 020075
3455 016210 020040 020040 020040
3456 016216 020040 041511 052116
3457 016224 020075
3458 016226 020040 020040 020040
3459 016234 000
3460 016235 040 046440 051120
3461 016242 020075
3462 016244 020040 020040 020040
3463 016252 020040 050115 020122
  
```

; ASCII MESSAGES

```

MEO:
MTNUM: .ASCII (15)(12)'PC= '
MPC: .ASCII ' ICNT= '
MICNT: .ASCIZ ' '
MSTR: .ASCII ' MPR= '
MTREG: .ASCII ' MPR DATA= '
  
```


3520	016676	047516	050040	051101	MT:	.ASCIZ	'NO PARITY MEMCRY FOUND'<15><12>
3521	016704	052111	020131	042515			
3522	016712	047515	054522	043040			
3523	016720	052517	042116	005015			
3524	016726	000					
3525	016727	116	020117	040520	MTR:	.ASCIZ	'NO PARITY REGSTER FOUND'<15><12>
3526	016734	044522	054524	051040			
3527	016742	043505	052123	051105			
3528	016750	043040	052517	042116			
3529	016756	005015	000				
3530	016761	040	020040	020040	MPRAD:	.ASCIZ	'<15><12>
3531	016766	020040	006440	000012			
3532	016774	006577	006412	046412	MTIT:	.ASCIZ	<177><15><12><15><12>'MEMORY PARITY TEST - MAINDEC-11-DCMFA-D'
3533	017002	046505	051117	020131			
3534	017010	040520	044522	054524			
3535	017016	052040	051505	020124			
3536	017024	020055	040515	047111			
3537	017032	042504	026503	030461			
3538	017040	042055	046503	040505			
3539	017046	042055	000				
3540	017051	015	050012	053517	MPWRF:	.ASCIZ	<15><12>'POWER FAILED'
3541	017056	051105	043040	044501			
3542	017064	042514	000104				
3543	017070	007			MPGEND:	.BYTE	007
3544	017071	015	042412	042116		.ASCII	<15><12>'END PASS = '
3545	017076	050040	051501	020123			
3546	017104	020075					
3547	017106	020040	020040	020040	MPCNT:	.ASCIZ	'<15><12>
3548	017114	000					
3549	017115	015	005015	040502	MPSER:	.ASCII	<15><15><12>'BAD PAR FOUND IN LCC '
3550	017122	020104	040520	020122			
3551	017130	047506	047125	020104			
3552	017136	047111	046040	041517			
3553	017144	040					
3554	017145	040	020040	020040	MPSER1:	.ASCIZ	'<15><12>
3555	017152	000040					
3556	017154	005015	042522	044507	MX1:	.ASCIZ	<15><12>'REGISTER AT '
3557	017162	052123	051105	040440			
3558	017170	020124	000				
3559	017173	103	047117	051124	MX2:	.ASCIZ	'CONTROLS '
3560	017200	046117	020123	000			
3561	017205	015	046012	040517	LDRMSG:	.ASCIZ	<15><12>'LOADERS RESTORED'
3562	017212	042504	051522	051040			
3563	017220	051505	047524	042522			
3564	017226	000104					
3565	017230	005015	040502	020104	PSMSG:	.ASCIZ	<15><12>'BAD PARITY SCAN COMPLETE'
3566	017236	040520	044522	054524			
3567	017244	051440	040503	020116			
3568	017252	047503	050115	042514			
3569	017260	042524	000				
3570	017263	015	046012	040517	MLDRSV:	.ASCII	<15><12>'LOADERS SAVED IN BANK 0'
3571	017270	042504	051522	051440			
3572	017276	053101	042105	044440			
3573	017304	020116	040502	045516			
3574	017312	030040					
3575	017314	005015	047524	051040		.ASCIZ	<15><12>'TO RESTORE LOADERS USE SA 210.'

3576	017322	051505	047524	042522
3577	017330	046040	040517	042504
3578	017336	051522	052440	042523
3579	017344	051440	020101	030462
3580	017352	027060	000	
3581	017355	040	020040	020040
3582	017362	020040	026440	041440
3583	017370	051117	020105	040520
3584	017376	044522	054524	051040
3585	017404	043505	051511	042524
3586	017412	006522	000012	
3587	017416	020040	020040	020040
3588	017424	020040	020055	047515
3589	017432	020123	040520	044522
3590	017440	054524	051040	043505
3591	017446	051511	042524	006522
3592	017454	000012		
3593				
3594	000001			

MPCOR: .ASCIZ ' - CORE PARITY REGISTER'<15><12>

MPMOS: .ASCIZ ' - MOS PARITY REGISTER'<15><12>

.EVEN
.END

ADRPT	000516	DONE14	005450	INDC7	001010	MCR	016671	MTYFG	000564
ADRS =	077400	DONE15	006114	INDC8	001012	MDASH	016666	MX1	017154
ADRTYP	001224	DONE16	006444	INDC9	001014	MDATA	016266	MX2	017173
ADTEMP	015406	DONE17	007360	INDX17=	007642	MEMH	001070	NOKT	000552
AE =	000001	DONE20	010334	INST1	004070	MEML	001066	NOP =	000240
BOCNV	015242	DONE4	003044	INST2	004076	MEMUT	000550	NOREG	002026
BOCNVA	015274	DONE5	003226	KPAR0	001244	MEO	016202	NRALL	013430
BOCNVB	015326	DONE6	003566	KPAR1	001246	MICNT	016226	OACDST	015236
BOCNVC	015330	DONE7	003700	KPAR2	001250	MK	016674	OACNT	015240
BOCNVD	015332	DSWR =	177570	KPAR7	001252	MLDRSV	017263	OACNV	015134
BEGIN	002052	EMTA	015574	KPDR0	001234	MMAOR	016457	OACNVA	015156
BITPT	000520	EMTINT	015550	KPDR1	001236	MMDEV	016424	OACNVX	015234
BIT0 =	000001	EMTLIM	015636	KPDR2	001240	MMPAT	016513	ODDFLG	000560
BIT1 =	000002	EMTTAB	015614	KPDR7	001242	MMPRS	016544	OPEN =	000000
BIT10 =	002000	EMTX =	000011	KTSTAR	001222	MPC	016210	PARPAT	001040
BIT11 =	004000	ERR	013300	LDRMSG	017205	MPCNT	017106	PARVEC=	000114
BIT12 =	010000	ERRA	013322	LDRSYD	014134	MPGEND	017070	PASCNT	000544
BIT13 =	020000	ERRA1	013350	LOCATM	013644	MPRAD	016761	PDREND	001232
BIT14 =	040000	ERRA2	013352	LOCAT1	013742	MPRCOR	017355	PDRTAB	001226
BIT15 =	100000	ERRA3	013362	LOGICA	011434	MPRMOS	017416	PERR =	100000
BIT2 =	000004	ERRA4	013364	LOOP10	003772	MPROK	000542	PMEMH	001074
BIT3 =	000010	ERRA5	013374	LOOP11	004322	MPRO	000570	PMEML	001072
BIT4 =	000020	ERRB	013376	LOOP12	004536	MPR1	000600	PMEMX	001076
BIT5 =	000040	ERRBX	013400	LOOP13	005030	MPR10	000710	PS =	177776
BIT6 =	000100	ERRC	013404	LOOP14	005306	MPR11	000720	PSADRS	014536
BIT7 =	000200	ERRD	013426	LOOP15	005740	MPR12	000730	PSCAN	014232
BIT8 =	000400	ERROR =	104002	LOOP21	011030	MPR13	000740	PSCANH	014540
BIT9 =	001000	ERRORP=	104003	LOOP4	002666	MPR14	000750	PSCANX	014500
BMOVA	015436	ERRORS=	104004	LOOP5	003122	MPR15	000760	PSCAN1	014542
BMOVE	015426	ERRP	013160	LOOP6	003276	MPR2	000610	PSCX1	014654
CKBKO	010350	ERRST	013102	LOOP7	003640	MPR3	000620	PSLOOP	014576
CLRPAR	013530	FTITLE	000512	LOP15	005750	MPR4	000630	PSLUP	014602
CNT16	007110	GMPRA	001630	LOP4	002674	MPR5	000640	PSMSG	017230
CNT17	007770	GMPRB	001756	LOP5	003130	MPR6	000650	PSNXT	014626
CNVCTR	015400	GMPRC	001764	LOP6	003304	MPR7	000660	PSXTST	014662
CN16	007130	GMPRD	002020	LOWFLG	000556	MPR8	000670	PWRUN	015450
CONT	011454	HIADR	000530	LSAV	014016	MPR9	000700	PWRUP	015514
CONT10	004200	HIWORD	000554	LUP10	003764	MPSER	017115	RELOC	000566
CONT12	004730	ICNT	016176	LUP11	004306	MPSER1	017145	RESRVD	001032
CONT13	005210	IMAX	016174	LUP12	004522	MPWRF	017051	RESVC	001034
CONT3	002522	INDCO	000772	LUP13	005022	MSETSR	016363	RESVM	001036
CONT4	003000	INDC1	000774	LUP14	005316	MSTR	016235	RETURN	016200
COUNT	004242	INDC10	001016	LUP7	003632	MSTRX	016277	RSTART	001272
DOISP =	177570	INDC11	001020	MAPMA	011526	MSTRX1	016323	RSTLDR	014136
DECVAL	015420	INDC12	001022	MAPMB	011720	MSTRX3	016337	RSTLDX	014226
DIGIT	015402	INDC13	001024	MAPMEM	011514	MSTRXS	016353	RSTO4 =	104006
DISPLA	001102	INDC14	001026	MAPRB	012104	MT	016676	RSTOSS=	104007
DISPRE	000174	INDC15	001030	MAPRC	012204	MTIT	016774	RSO4	015726
DONE	011374	INDC2	000776	MAPRD	012304	MTMAP	016612	RSOSS	015754
DONE10	004226	INDC3	001000	MAPRE	012400	MTNUM	016202	ROSAV	001256
DONE11	004446	INDC4	001002	MAPREG	012042	MTR	016727	RISAV	001260
DONE12	004744	INDC5	001004	MAP1	013600	MTREG	016244	R2SAV	001262
DONE13	005224	INDC6	001006	MCON	016404	MTYCOR	016661	R3SAV	001264

RMSAV	001266	SV04	015640	TEST3	002470	TRP15	006234	TYPE	=	104000
RSSAV	001270	SV05A	015670	TEST4	002616	TRP16	006672	TYPESAX	015120	
R6	=X000006	SV05B	015702	TEST5	003064	TRP16A	006676	TYPESBX	015130	
SAVLDR	014020	SV05C	015716	TEST6	003240	TRP17	007544	TYPESX	015102	
SAVLDX	014116	SV05S	015660	TEST7	003600	TRP17A	007550	WAS	000536	
SAV04	= 104005	SWR	001100	TMAP	012556	TRP20	010470	WWP	=	000004
SAVOSS	= 104010	SWREG	000176	TMAPA	012572	TRP4A	003014	WWP16	006462	
SCAN	001352	TBANK	000546	TMAPB	012676	TRP4B	003022	WWP16A	006500	
SCANA	001374	TEMP	000562	TMAPC	012764	TRP4C	003030	WWP17	007376	
SCANB	001400	TEMPX	000514	TMAPD	013030	TRP7	003710	WWP17A	007414	
SCNFLG	001220	TENPWR	015404	TMAPEX	013100	TSTLOC	000532	WWP21	010660	
SCOPE	= 104001	TESTX	016010	TNUM	=	000022	TSTX	000510	WWP21A	010674
SCOPEB	016144	TEST1	002074	TPB	001106	TST10	004010	XFR1	010034	
SCOPEC	016074	TEST10	003722	TPCORE	005460	TST11	004340	XFR2	011270	
SCOPEG	016156	TEST11	004244	TPCORX	006130	TST12	004554	SENDAD	011444	
SHOBE	000534	TEST12	004460	TPS	001104	TST13	005044	SFILLS	001111	
SPSAV	001254	TEST13	004764	TRDATA	000540	TST14	005342	SNULL	001110	
SRO	= 177572	TEST14	005244	TREG	000770	TST15	006014	STPFLG	001112	
START	001432	TEST15	005646	TRFLG	000522	TST4	002706	STYPE	001114	
START1	001576	TEST16	006314	TRP10	004106	TST5	003144	.	=	017456
STKPT	= 000510	TEST17	007154	TRP10A	004156	TST6	003320			
SUBTEN	015340	TEST2	002246	TRP12	004630	TST7	003654			
SUBTNA	015344	TEST20	010222	TRP13	005120	TYCOR	000526			
SUBTNB	015360	TEST21	010624	TRP14	005566	TYFLG	000524			

. ABS. 017456 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DSKZ:DCMFAD, DSKZ:DCMFAD/SOL=DSKZ:DCMFAD.P11
RUN-TIME: 12 27 1 SECONDS
RUN-TIME RATIO: 726/41=17.6
CORE USED: 6K (11 PAGES)