

.XEROX COPIES=02, FORMS=NONE, END;

CCMFAFO, MEMORY PARI
SEQ 0001

TY TEST MACY11 30A(1052) 13-JAN-78 12:27 PAGE 1
CCMFAP.P11 13-JAN-78 12:13

.REM *

IDENTIFICATION

PRODUCT CODE: AC-7E93F-MC
PRODUCT NAME: CCMFAFO MS11, MF11, MA11-P MEM
DATE RELEASED: FEB 1978
MAINTAINER: DIAGNOSTIC GROUP

COPYRIGHT (C) 1973, 1978
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, OR XXDP.

4.0 STARTING PROCEDURE

4.1 STARTING ADDRESSES

200= NORMAL (WORST CASE) TESTING
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).

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

SW 08=1 OR UP -- THAT WAS NOT LOCATED)
-- 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 TIMEOUT ROUTINE)

THIS IS THE STANDARD TIMEOUT ROUTINE, ALLOWING PATCHING TO UTILIZE OUTPUT DEVICES OTHER THAN THE ASR 33. \$NULL CONTAINS THE VALUE TC 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 3 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=ZZZZZ PC OF FAILING ERROR CALL. REFER TO THIS ADDRESS IN THE LISTING FOR AN EXPLANATION OF THE ERROR.

ICNT=YYYYY 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 11. KERNEL PAGE ADDRESS REGISTER 1 (ADDRESS 775340)

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 ZEROS (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 KT11 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 ZEROS 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.
XXDP CHAINING IS POSSIBLE FOR SYSTEMS GREATER THAN 8K.

7.1 STARTING PROCEDURE

PROGRAM MUST BE LOADED INTO LOWER 4K OF MEMO:Y.

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(INDC0-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

325
326
327
328
329
330 ;SWITCH REGISTER SWITCH OPTIC IS (SWITCH SET TO A 1)
331 ;SR15 - HALT ON ERROR
332 ;SR14 - SCOPE
333 ;SR13 - INHIBIT PRINTOUT
334 ;SR11 - INHIBIT ITERATIONS
335 ;SR10 - HALT AFTER LOCATING BAD PARITY BEFORE CORRECTING IT
336 ;SR09 - HALT AFTER TYPING PARITY MEMORY MAP (ALLOWS MANUAL
337 ; CHANGES TO BE MADE TO THE MAP TO FORCE TESTING OF
338 ; MEMORY THAT WAS NOT LOCATED)
339 ;SR08 - HALT AT END OF PASS (IF HALTED ELSEWHERE, THE PROGRAM
340 ; MAY BE RELOCATED TO BANK1, WRITE WRONG PARITY MAY BE SET,
341 ; AND/OR BAD PARITY MAY EXIST IN THE PARITY MEMORY).
342
343
344
345 ;SYMBOL DEFINITIONS
346 000001 BIT0=1 ;BIT DEFINITIONS
347 000002 BIT1=2
348 000004 BIT2=4
349 000010 BIT3=10
350 000020 BIT4=20
351 000040 BIT5=40
352 000100 BIT6=100
353 000200 BIT7=200
354 000400 BIT8=400
355 001000 BIT9=1000
356 002000 BIT10=2000
357 004000 BIT11=4000
358 010000 BIT12=10000
359 020000 BIT13=20000
360 040000 BIT14=40000
361 100000 BIT15=100000
362 000001 AE=1 ;ACTION ENABLE
363 000004 WWP=4 ;WRITE WRONG PARITY
364 077400 ADRS=77400 ;ADDRESS OF ERROR
365 100000 PERR=100000 ;PARITY ERROR BIT
366 177570 DSWR=177570 ;HARDWARE SWITCH REGISTER
367 177570 DDISP=177570 ;HARDWARE DISPLAY REGISTER
368 177776 PS=177776
369 000007 PC=%7
370 000006 SP=%6
371 000240 NOP=240
372 000000 OPEN=0
373 000510 STKPT=TSTX
374 000000 R0=%0
375 000001 R1=%1
376 000002 R2=%2
    
```

```

377 000003 R3=%3
378 000004 R4=%4
379 000005 R5=%5
380 000006 R6=%6
381 000114 PARVEC=114 ;PARITY ERROR TRAP VECTOR
382 177572 SRO=177572 ;ADDRESS OF MEM MGMT REGISTER SRO
383
384
385
386
387 ;MACRO DEFINITIONS
388
389
390
391
392
393 ;TRAPCATCHER (.+ ,HALT) LOADED INTO LOCATIONS 000-576
394
395 ;LOAD EMT VECTOR
396 . =30
397 000030 016032 EMTINT
398 000032 000340 340
399 000046 . =46
400 000046 011702 $ENDAD
401 000052 . =52
402 000052 040000 BIT14
403
404 ;LOAD STARTING ADDRESS AREA
405 . =174
406 000174 000000 DISPREG: .WORD 0 ;THIS IS THE SOFTWARE DISPLAY REGISTER
407 000176 000000 SWREG: .WORD 0 ;THIS IS THE SOFTWARE SWITCH REGISTER
408 000200 000167 001232 JMP START ;GO TO START OF PROGRAM
409 000210 000210 . =210
410 000210 000167 014204 JMP RSTLDR ;GO RESTORE THE LOADERS
411 000220 000220 . =220
412 000220 000167 001132 JMP SCAN ;SCAN FOR BAD PARITY
413 000230 000230 . =230
414 000230 000167 001042 JMP RSTART ;RESTART WITHOUT RETYPING MAP INFORMATION
415
416 000510 . =510
417
418
419
420
421 ;GENERAL DATA AREA
422 000510 000000 TSTX: 0
423 000512 000000 FTITLE: 0 ;TITLE PRINTED = 1
424 000514 000000 TEMPX: 0
425 000516 000000 ADRPT: 0 ;MAPPING- ADDRESS POINTER
426 000520 000000 BITPT: 0 ;MAPPING- BIT POINTER INDICATING BANK
427 000522 000000 TRFLG: 0 ;MAPPING- TRANSITION FLAG
428 000524 000000 TYFLG: 0 ;MAPPING- TYPED FLAG
429 000526 000000 TYCOR: 0 ;MAPPING- K CORE ACCUMULATOR
430 000530 000000 HIADR: 0 ;USED TO CHECK WHEN DONE TESTING A BANK
431 000532 000000 TSTLOC: 0 ;LOADED WITH ADDRESS OF LOCATION UNDER
432 ;TEST IN SOME SUBTESTS
    
```

```
433 000534 000000 SHDBE: 0 ;VALUE EXPECTED
434 000536 000000 WAS: 0 ;ACTUAL VALUE FOUND
435 000540 000000 TRDATA: 0
436 000542 000000 MPROK: 0
437 000544 000000 TBANK: 0
438 000546 000000 MEMUT: 0
439 000550 000000 NQKT: 0 ;SET TO INDICATE NO KT11 PRESENT
440 000552 000000 HIWORD: 0
441 000554 000000 LOWFLG: 0
442 000556 000000 ODDFLG: 0 ;IF SET INDICATES TESTING HIGH BYTE
443 ;OF MEMORY LOCATION
444 000560 000000 TEMP: 0
445 000562 000000 MTYFG: 0 ;SET TO INDICATE MAP OF PARITY MEMORY
446 ;ALREADY TYPED
447 000564 000000 RELOC: 0
448
449
450
451 ;MEMORY PARITY CONTROL REGISTER ADDRESSES
452 ;THE LEAST SIGNIFICANT BIT IN THE DEVICE ADDRESS IS SET TO A ONE(1)
453 ;IF THE CONTROL IS FOUND NOT TO BE PRESENT. THE MEMORY PRESENT UNDER
454 ;CONTROL OF EACH CONTROLLER IS REPRESENTED BY 2 OCTAL WORDS. EACH BIT
455 ;REPRESENTS A 4K BLOCK, I.E. BIT0= 0-4K, BIT1= 4-8K, BIT15= 60-64K.
456 ;THE LOW BYTE OF THE LAST WORD FOR EACH REGISTER INDICATES THE OFFSET (0,2,4,OR 6)
457 ;FOR THE FIRST ADDRESS THAT ACTUALLY CORRESPONDED TO THE REGISTER. THE HIGH BYTE GETS
458 ;SET TO 1 TO INDICATE THAT A MEMORY ADDRESS HAS BEEN FOUND FOR THAT REGISTER.
459 ;FOR EXAMPLE, SAY THAT MPRO AND MPR1 EXIST, CONTROLLING INTERLEAVED MEMORY
460 ;FROM 0 TO 16K, AND THAT MPRO CONTROLS THE ADDRESSES ENDING IN 0 AND 4.
461 ;THE MAP WOULD THEN LOOK AS FOLLOWS:
462 ; MPRO: 172100 ;BIT 0 IS CLEAR SINCE REGISTER IS PRESENT
463 ; 17 ;REGISTER CONTROLS 1ST 16K (=4 BANKS)
464 ; 0
465 ; 400 ;LOW BYTE SHOWS THAT FIRST ADDRESS
466 ; ;ENDS IN 0 (OCTAL)
467 ; ;HIGH BYTE CONTAINS A 1 TO INDICATE
468 ; ;THAT AN ADDRESS WAS FOUND
469 ; MPR1: 172102 ;BIT 0 IS CLEAR SINCE REGISTER IS PRESENT
470 ; 17 ;REGISTER CONTROLS 1ST 16K
471 ; 0
472 ; 402 ;LOW BYTE INDICATES THAT THE FIRST
473 ; ;MEMORY ADDRESS ENDS IN 2 (OCTAL)
474 ; ;HIGH BYTE CONTAINS A 1 TO INDICATE
475 ; ;THAT AN ADDRESS WAS FOUND
476 ;THE REST OF THE MAP WOULD APPEAR AS IN THE LISTING
477
478 000566 172101 MPRO: 172100+1 ;PARITY STATUS REGISTERS
479 000570 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
480 000572 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
481 000574 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
482 000576 172103 MPR1: 172102+1
483 000600 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
484 000602 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
485 000604 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
486 000606 172105 MPR2: 172104+1
487 000610 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
488 000612 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
```

```
489 000614 000000 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
490 000616 172107 MPR3: 172106+1
491 000620 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
492 000622 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
493 000624 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
494 000626 172111 MPR4: 172110+1
495 000630 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
496 000632 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
497 000634 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
498 000636 172113 MPR5: 172112+1
499 000640 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
500 000642 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
501 000644 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
502 000646 172115 MPR6: 172114+1
503 000650 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
504 000652 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
505 000654 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
506 000656 172117 MPR7: 172116+1
507 000660 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
508 000662 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
509 000664 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
510 000666 172121 MPR8: 172120+1
511 000670 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
512 000672 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
513 000674 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
514 000676 172123 MPR9: 172122+1
515 000700 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
516 000702 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
517 000704 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
518 000706 172125 MPR10: 172124+1
519 000710 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
520 000712 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
521 000714 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
522 000716 172127 MPR11: 172126+1
523 000720 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
524 000722 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
525 000724 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
526 000726 172131 MPR12: 172130+1
527 000730 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
528 000732 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
529 000734 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
530 000736 172133 MPR13: 172132+1
531 000740 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
532 000742 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
533 000744 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
534 000746 172135 MPR14: 172134+1
535 000750 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
536 000752 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
537 000754 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
538 000756 172137 MPR15: 172136+1
539 000760 000000 0 ;0-64K PARITY MEM UNDER THIS CONTROL
540 000762 000000 0 ;64-124K PARITY MEM UNDER THIS CONTROL
541 000764 000000 0 ;ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
542
543 000766 000000 TREG: 0 ;PARITY REGISTER UNDER TEST
544
```



```

545
546
547 ;INDICATORS FOR CORE OR MOS PARITY REGISTER:
548 ;EACH INDICATOR REFERS TO A PARTICULAR PARITY REGISTER. IF IT IS
549 ;A CORE PARITY REGISTER THEN A '1' IS STORED IN THE INDICATOR.
550 ;IF IT IS A MOS PARITY REGISTER THEN '-1' GETS STORED.
551 ;EX= IF MPRO (172100) IS FOR CORE AND MPR1 (172102) IS FOR MOS
552 ;THEN THE INDICATOR MAP WILL LOOK AS FOLLOWING:
553 ;INDC0: 000001
554 ;INDC1: 177777
555
556 000770 000000 INDCC: 0 ;CORE-MOS PARITY INDICATOR FOR M'3
557 000772 000000 INDCC1: 0 ;CORE-MOS PARITY INDICATOR FOR MPR1
558 000774 000000 INDCC2: 0 ;CORE-MOS PARITY INDICATOR FOR MPR2
559 000776 000000 INDCC3: 0 ;CORE-MOS PARITY INDICATOR FOR MPR3
560 001000 000000 INDCC4: 0 ;FOR MPR4
561 001002 000000 INDCC5: 0 ;FOR MPR5
562 001004 000000 INDCC6: 0 ;FOR MPR6
563 001006 000000 INDCC7: 0 ;FOR MPR7
564 001010 000000 INDCC8: 0 ;FOR MPR8
565 001012 000000 INDCC9: 0 ;FOR MPR9
566 001014 000000 INDCC10: 0 ;FOR MPR10
567 001016 000000 INDCC11: 0 ;FOR MPR11
568 001020 000000 INDCC12: 0 ;FOR MPR12
569 001022 000000 INDCC13: 0 ;FOR MPR13
570 001024 000000 INDCC14: 0 ;FOR MPR14
571 001026 000000 INDCC15: 0 ;FOR MPR15
572 001030 000000 RESRVD: 0
573
574 ;BIT POSITIONS WHICH ARE RESERVED
575 001032 070032 RESVC: 70032 ;FOR FUTURE USE IN PARITY REGISTERS
576 001034 077772 RESVM: 77772 ;CORE PARITY
577 ;MOS PARITY
578
579
580
581 ;PARITY PATTERNS
582 001036 125325 PARPAT: 125325 ;EVEN, ODD BYTES
583 001040 152652 ;ODD, EVEN
584 001042 052452 ;EVEN, ODD
585 001044 025125 ;ODD, EVEN
586 001046 102070 ;EVEN, EVEN
587 001050 072527 ;ODD, ODD
588 001052 177777 ;EVEN, EVEN
589 001054 107030 ;ODD, ODD
590 001056 152525 ;ODD, EVEN
591 001060 000000 0 ;EXTRA PATTERN AREA
592 001062 000000 0 ;TERMINATOR, DO NOT USE THIS LOC
593
594
595 ;THIS IS A MAP OF THE TOTAL MEMORY PRESENT IN THE SYSTEM.
596 001064 000000 MEML: 0 ;0-64K MEM PRESENT IN 4K CONTIGUOUS BLOCKS
597 001066 000000 MEMH: 0 ;64-124 MEM PRESENT IN 4K CONTIGUOUS BLOCKS
598
599
600 ;THIS IS A MAP OF THE TOTAL PARITY MEMORY PRESENT IN THE SYSTEM.
600 001070 000000 PMEML: 0 ;0-64K PARITY MEMORY PRESENT

```

```

601 ;(IN 4K CONTIGUOUS BLOCKS)
602 001072 000000 PMEMH: 0 ;64-124K PARITY MEMORY PRESENT
603 ;(IN 4K CONTIGUOUS BLOCKS)
604 001074 000000 PMEMX: 0 ;TEMP TO HOLD CONTENTS OF EITHER
605 ;LOW OR HIGH MAP
606
607
608
609 ;ROUTINE TO TYPE ASCII MESSAGES, MESSAGE MUST TERMINATE WITH A 0 BYTE.
610 ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
611 ;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
612 ;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
613
614 ;=1100
615 001100 177570 SWR: .WORD DSWR ;ADDRESS OF THE SWITCH REGISTER
616 001102 177570 DISPLAY: .WORD DDISP ;ADDRESS OF THE DISPLAY REGISTER
617 001104 177564 TPS: 177564 ;PRINTER STATUS REGISTER ADDRESS
618 001106 177566 TPB: 177566 ;PRINTER BUFFER REGISTER ADDRESS
619 001110 000 $NULL: .BYTE 0 ;CONTAINS NULL CHARACTER FOR FILLS
620 001111 002 $FILLS: .BYTE 2 ;CONTAINS # OF FILLER CHARACTERS REQUIRED
621 001112 000 $TPFLG: .BYTE 0 ;"TERMINAL AVAILABLE" FLAG (0=YES)
622 001113 000 ;RESERVED
623
624 001114 105767 177772 $TYPE: TSTB $TPFLG ;IS THERE A TERMINAL?
625 001120 001401 BEO 6$ ;BR IF YES
626 001122 000000 HALT ;HALT HERE IF NO TERMINAL
627 001124 010046 6$: MOV R0, -(SP) ;SAVE R0
628 001126 017600 000002 MOV @2(SP), R0 ;GET ADDRESS OF ASCII STRING
629 001132 112046 1$: MOVB (R0)+, -(SP) ;PUSH CHARACTER TO BE TYPED ONTO STACK
630 001134 001005 BNE 2$ ;BR IF IT ISN'T THE TERMINATOR
631 001136 005726 TST (SP)+ ;IF TERMINATOR POP IT OFF THE STACK
632 001140 012600 MOV (SP)+, R0 ;RESTORE R0
633 001142 062716 000002 7$: ADD #2, (SP) ;ADJUST RETURN PC
634 001146 000002 RTI ;RETURN
635 001150 004767 000026 2$: JSR PC, 5$ ;GO TYPE THIS CHARACTER
636 001154 122726 000012 3$: CMPB #12, (SP)+ ;CHECK IF THE CHARACTER TYPED
637 ;WAS A LINE FEED
638 001160 001364 BNE 1$ ;GO GET NEXT CHARACTER IF NOT LINE FEED
639 001162 016746 177722 MOV $NULL, -(SP) ;GET # OF FILLER CHARACTERS NEEDED
640 ;AND THE NULL CHARACTER
641 001166 105366 000001 4$: DECB 1(SP) ;DOES A NULL NEED TO BE TYPED?
642 001172 002770 BLT 3$ ;BR IF NO--GO POP THE NULL OF THE STACK
643 001174 004767 000002 JSR PC, 5$ ;GO TYPE A NULL
644 001200 000772 BR 4$ ;LOOP
645 001202 105777 177676 5$: TSTB @TPS ;WAIT UNTIL PRINTER IS READY
646 001206 100375 BPL 5$
647 001210 116677 000002 177670 MOVB 2(SP), @TPB ;LOAD CHARACTER TO BE
648 ;TYPED INTO DATA REGISTER
649 001216 000207 RTS PC
650
651
652 ;GENERAL DATA AREA
653
654
655 001220 000000 SCNFLG: 0 ;SCNFLG GETS SET IF USING
656 ;SCAN ROUTINE (SA=220)

```

```

657 001222 000000          CACHFL: 0          ;CACHE FLAG
658 001224 177746          CACHE: 177746       ;CACHE CSR
659 001226 000000          KTSTART: 0
660 001230 000000          ADRTYP: 0
661 001232 177600          PDRTAB: 177600
662 001234 172200          172200
663 001236 172300          PDREND: 172300
664 001240 172300          KPDR0: 172300       ;KERNEL PAGE DESCRIPTOR REGISTER ADDRESSES
665 001242 172302          KPDR1: 172302
666 001244 172304          KPDR2: 172304
667 001246 172316          KPDR7: 172316
668 001250 172340          KPAR0: 172340       ;KERNEL PAGE ADDRESS REGISTER ADDRESSES
669 001252 172342          KPAR1: 172342
670 001254 172344          KPAR2: 172344
671 001256 172356          KPAR7: 172356
672 001260 000000          SPSAV: 0           ;REGISTER SAVE LOCATIONS
673 001262 000000          ROSAV: 0
674 001264 000000          R1SAV: 0
675 001266 000000          R2SAV: 0
676 001270 000000          R3SAV: 0
677 001272 000000          R4SAV: 0
678 001274 000000          R5SAV: 0
679
680
681
682
683 001276 012706 000510          ;ROUTINE TO RESTART WITHOUT RETYPING MAP AFTER TEST HAS BEEN RUNNING
684 001302 012767 000001          RSTART: MOV #STKPT,SP ;SET UP STACK POINTER
685 001310 005067 010436          177252 MOV #1,MTYFG ;SET FLAG TO INDICATE MAP HAS BEEN TYPED
686 001314 012737 015732          000024 CLR PASCNT ;INITIALIZE PASS COUNT
687 001322 012737 000340          000026 MOV #PWRDN,@#24
688 001330 005067 177154          CLR #340,@#26
689 001334 005037 177776          CLR TSTX
690 001340 012737 000006          000004 CLR @#P5 ;CLEAR PROCESSOR STATUS REGISTER
691 001346 005037 000006          MOV #6,@#4
692 001352 000167 000530          CLR @#6
693          JMP BEGIN
694
695
696
697
698          ;ROUTINE TO SCAN ALL MEMORY FOR BAD PARITY AND TYPE 18 BIT ADDRESSES OF BAD
699 001356 012706 000510          ;LOCATIONS
700 001362 005767 177124          SCAN: MOV #STKPT,SP ;SETUP STACK POINTER
701          TST FTITLE ;IF TITLE HAS BEEN PRINTED, REGISTERS
702          ;HAVE ALREADY BEEN LOCATED- GO
703          BNE SCANB ;AND LOCATE IF NOT ALREADY DONE
704          INC SCNFLG ;BRANCH, REGISTERS HAVE ALREADY BEEN LOCATED
705          JMP START1 ;INCREMENT SCNFLG
706          CLR SCNFLG ;GO TO LOCATE THE REGISTERS
707          JSR PC,MAPMEM ;RETURN HERE AFTER LOCATING THE REGISTERS
708          JSR PC,PSCAN ;SETUP MEMORY MAP
709          TYPE ;SCAN FOR BAD PARITY
710          PMSMSG ;TYPE MESSAGE "BAD PARITY SCAN COMPLETE"
711          TST NOKT
712          BNE .+6
    
```

```

713 001426 005037 177572          CLR @#SRO          ;TURN OFF KT11 IF PRESENT
714 001432 000000          HALT              ;END OF PARITY SCAN
715 001434 000750          BR SCAN
716
717
718
719
720          ;NORMAL STARTUP
721 001436 012706 000510          START: MOV #STKPT,SP ;SET UP STACK POINTER
722 001442 005067 177114          CLR MTYFG ;CLEAR FLAG WHICH INDICATES MAP TYPED
723 001446 005067 010300          CLR PASCNT ;INITIALIZE PASS COUNT
724 001452 012737 015732          000024 MOV #PWRDN,@#24 ;SETUP POWER FAIL RETURN
725 001460 012737 000340          000026 MOV #340,@#26
726 001466 013746 000004          MOV @#4,-(SP) ;SAVE THE ERROR VECTOR
727 001472 013746 000006          MOV @#6,-(SP)
728 001476 012737 001512          000004 MOV #2$,@#4 ;
729 001504 005777 177370          TST @#SWR ;SET UP TIME OUT VECTOR
730 001510 000407          BR 4$ ;TRY TO REFERENCE THE SWITCH REGISTER
731 001512 012767 000176 177360 2$: MOV #SWREG,SWR ;BRANCH IF NO TIME OUT
732 001520 012767 000174 177354 MOV #DISPREG,DISPLAY ;POINT TO THE SOFTWARE SWITCH REGISTER
733 001526 022626          CMP (SP)+,(SP)+ ;POINT TO THE SOFTWARE DISPLAY REG.
734 001530 012737 001562 000004 4$: MOV #6$,@#4 ;RESTORE THE STACK POINTER
735 001536 012737 000340 000006 MOV #340,@#6
736 001544 052737 000014 177746 BIS #14,@#177746 ;DISABLE CACHE
737 001552 052767 000200 177442 BIS #200,CACHFL ;SET FLAG FOR CACHE
738 001560 000401          BR 5$
739 001562 022626          6$: CMP (SP)+,(SP)+
740 001564 012637 000006          5$: MOV (SP)+,@#6 ;RESTORE ERROR VECTOR
741 001570 012637 000004          MOV (SF)+,@#4
742 001574 005067 176710          CLR TSTX
743 001600 005767 176706          1$: TST FTITLE ;IS TITLE PRINTED YET?
744 001604 001012          BNE START1 ;YES, SKIP OVER
745 001606 005267 176700          INC FTITLE ;SET FLAG
746 001612 023737 000042 000046 CMP @#42,@#46 ;ARE WE IN ACT11 AUTOMATIC MODE?
747 001620 001404          BEQ START1 ;YES, SKIP TITLE
748 001622 104000          TYPE ;TYPE TITLE "MEMORY PARITY TEST
749 001624 017257          MTIT ;MAINDEC-11-DCMFA"
750 001626 104000          TYPE
751 001630 017636          MLDRSV ;TYPE "LOADERS SAVED IN BANK 1."
752
753
754
755
756          ;SEARCH FOR PARITY REGISTERS PRESENT AND TYPE ADDRESSES OF THOSE FOUND
757          ;FAILURE TO LOCATE A REGISTER INDICATES THAT THE ADDRESS TIMED OUT OR THAT
758          ;BITS 5-7 IN THE REGISTER DID NOT SET
759 001632 104000          STAR1: TYPE ;TYPE "MEMORY PARITY REGISTERS PRESENT ARE:"
760 001634 017026          MMPRS
761 001636 005067 176700          CLR MPROK ;CLEAR MPR FLAG
762 001642 012702 000566          MOV #MPRO,R2 ;SET UP POINTERS
763 001646 012703 000770          MOV #INDCO,R3 ;POINTER TO CORE-MOS
764 001652 012737 002012          000004 MOV #GMPRB,@#4 ;SET UP TIMEOUT TRAP RETURN
765 001660 005037 000006          CLR @#6
766 001664 042712 000001          GMPRA: BIC #1,(2) ;CLEAR FLAG BIT IN TABLE
767 001670 005062 000002          CLR 2(R2) ;INITIALIZE LOCATIONS IN THE TABLE
768 001674 005062 000004          CLR 4(R2)
    
```

769 001700 005062 000006 CLR 6(R2)
770 001704 005772 000000 TST @ (2)
771 001710 052772 000340 000000 BIS #340,@(2) ;DOES THIS MPR EXIST? (IF NO, TIMES OUT)
772 001716 032772 000340 000000 BIT #340,@(2) ;YES- IS IT AN MF11-LP OR MA11-P CORE PARITY REG
773 001724 001414 BEQ 1\$
774 001726 011267 176562 MO: (2),TEMPX ;NO, IS IT A MOS-11 PARITY REGISTER? BRANCH
775 001732 004567 013460 JSR R5,DACNV ;YES- PRINT REGISTER ADDRESS
776 001736 000514 TEMPX ;(GET ASCII)
777 001740 017670 MPRCOR
778 001742 000006 6
779 001744 104000 TYPE ;(TYPE ADDRESS)
780 001746 017670 MPRCOR
781 001750 012713 000001 MOV #1,(R3) ;SET INDICATOR FOR CORE PARITY
782 001754 000413 BR 2\$
783 001756 011267 176532 1\$: MOV (2),TEMPX ;IT IS A MOS REGISTER, PRINT ADDRESS
784 001762 004567 013430 JSR R5,DACNV ;(GET ASCII)
785 001766 000514 TEMPX
786 001770 017731 MPRMOS 6
787 001772 000006 TYPE ;(TYPE ADDRESS)
788 001774 104000 MPRMOS
789 001776 017731 MOV #-(R3) ;SET INDICATOR FOR MOS PARITY
790 002000 012713 177777 INC MPRK ;SET MPR REGISTER PRESENT FLAG
791 002004 005267 176532 2\$: MPRK ;SKIP NEXT
792 002010 000403 BR GMPRC
793 002012 022626 GMPRB: CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
794 002014 052712 000001 BIS #1,@R2 ;SET FLAG INDICATING REGISTER NOT PRESENT
795 002020 062702 000010 GMPRC: ADD #10,R2 ;UPDATE POINTER
796 002024 005723 TST (R3)+
797 002026 020227 CMP R2,#TREG ;DONE YET?
798 002032 002714 BLT GMPRA ;NO, LOOP
799 002034 012737 000004 000004 MOV #6,@#4 ;YES, RESTORE TRAPCATCHER
800 002042 005767 177152 TST SCNFLG ;ARE YOU IN THE ROUTINE TO SCAN
801 BEQ GMPRD ;MEMORY FOR BAD PARITY-(SCAN)
802 002046 001402 JMP SCANA ;NO,BRANCH TO CARRY ON NORMALLY
803 002050 000167 177324 ;YES, GO BACK TO THE MEMORY
804 ;SCAN ROUTINE
805 002054 005767 176462 GMPRD: TST MPRK ;ANY PARITY REGISTERS PRESENT?
806 002060 001012 BNE BEGIN ;YES- GO TEST CONTROLS PRESENT
807 002062 104000 NOREG: TYPE ;NO- TYPE "NO PARITY REGISTER FOUND"
808 002064 017212 MTR
809 002066 005737 000042 TST @#42
810 002072 001402 BEQ .+6 ;LOADED BY MONITOR?
811 002074 000167 007572 JMP LOGICAL ;NO, BRANCH
812 002100 000000 HALT ;YES- EXIT, NO REGISTERS PRESENT
813 002102 000167 177330 JMP START ;NO REGISTERS TO TEST
814 ;IF CONTINUED, TRY AGAIN
815 002106 012767 002156 014346 BEGIN: MOV #TEST1+2,RETURN ;SETUP SCOPE RETURN
816 002114 105767 177102 TSTB CACHFL ;CACHE ?
817 002120 001403 BEQ .+10 ;BRANCH IF NO
818 002122 012777 000014 177074 MOV #14,@CACHE ;DISABLE CACHE
819 002130 004767 012146 JSR PC,SAVLDR ;SAVE MONITOR
820 002134 012767 000100 014314 MOV #100,IMAX ;MAXIMUM ITERATION COUNT
821 002142 012737 000006 000004 MOV #6,@#4 ;RESTORE TRAPCATCHER IN TIMEOUT VECTOR
822 002150 005067 176344 CLR BITPT
823
824

825 ;*****
826 ;SHOW THAT BITS 0,2,5-11, AND 15 OF EACH CORE PARITY REGISTER
827 ;CAN BE SET AND CLEARED. BITS 0,2,15 OF EACH MOS PARITY REGISTER
828 ;PRESENT CAN BE SET AND CLEARED
829 ;*****
830 TEST1: SCOPE
831 002154 104001 MOV #1,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
832 002156 012777 000001 176716 MOV #MPRO,R0 ;LOAD ADDRESS OF TABLE INTO R0
833 002164 012700 000566 MOV #INDCO,R4 ;LOAD ADDRESS OF INDICATOR IN R4
834 002170 012704 000770 1\$: BIT #1,@R0 ;IS THIS REGISTER PRESENT?
835 002174 032710 000001 BNE 4\$;NO- BRANCH TO GET NEXT ADDRESS
836 002200 001042 MOV @R0,R1 ;YES- LOAD R1 WITH ADDRESS OF
837 002202 011001 ;PARITY REGISTER
838 ;IS THIS REGISTER CORE?
839 002204 022714 000001 CMP #1,(R4) ;YES, CORE. STORE RESERVED BITS
840 002210 001004 BNE 5\$;NO
841 002212 016767 176614 176610 MOV RESVC,RESRVD ;MOS. STORE RESERVED BITS
842 002220 000403 BR .+10 ;LOAD R2 WITH VALUE OF FIRST BIT
843 002222 016767 176606 5\$: MOV RESVM,RESRVD ;TO BE TESTED
844 002230 012702 000001 MOV #1,R2 ;INITIALIZE PARITY REGISTER
845 CLR @R1 ;READ CONTENTS OF PARITY REGISTER
846 002234 005011 MOV @R1,TREG ;CLEAR BITS WHICH ARE RESERVED
847 002236 011167 176524 BIC RESRVD,TREG ;CHECK OTHER BITS- BRANCH IF OK
848 002242 046767 176562 176516 BEQ .+4 ;CLEAR INSTRUCTION DID NOT INITIALIZE
849 002250 001401 ERROR ;ALL USED BITS IN PARITY REGISTER
850 002252 104002 ;TO ZERO (R1 CONTAINS ADDRESS OF
851 ;FAILING REGISTER)
852 ;IS THIS BIT RESERVED?
853 854 002254 030267 176550 2\$: BIT R2,RESRVD ;YES- DON'T TEST IT SINCE IT
855 002260 001010 BNE 3\$;MAY BE ZERO OR ONE
856 ;NO- SET THIS BIT IN THE PARITY REGISTER
857 002262 010211 MOV R2,@R1 ;READ AND SAVE CONTENTS OF PARITY REGISTER
858 002264 011103 MOV @R1,R3 ;CLEAR PARITY REGISTER
859 002266 005011 CLR @R1 ;CLEAR BIT LOCATIONS THAT ARE
860 002270 046703 176534 BIC RESRVD,R3 ;RESERVED
861 ;CHECK REST
862 002274 020203 CMP R2,R3 ;BRANCH IF OK
863 002276 001401 BEQ .+4 ;PARITY REGISTER WHOSE ADDRESS IS IN R1
864 002300 104002 ERROR ;WAS INCORRECT AFTER THE VALUE IN R2
865 ;WAS WRITTEN INTO IT. ACTUAL CONTENTS
866 ;(WITH UNUSED BITS CLEARED) IS IN R3
867 ;ROTATE BIT TO BE TESTED
868 002302 006302 3\$: ASL R2 ;IF NOT DONE WITH ALL BIT POSITIONS
869 002304 103363 BCC 2\$;GO TEST THIS ONE
870 ;MOVE R0 TO POINT TO NEXT POSSIBLE ADDRESS
871 002306 062700 000010 4\$: ADU #10,R0
872 002312 005724 TST (R4)+
873 ;OF A PARITY REGISTER
874 002314 020027 000766 CMP R0,#TREG ;AT END OF TABLE?
875 002320 002725 BLT 1\$;NO, BRANCH
876 002322 005067 176440 CLR TREG
877 ;*****
878 ;SHOW THAT RESET CLEARS BITS 0,2, AND 15 OF EACH PARITY REGISTER
879
880

```

;PRESENT.
;*****
TEST2: SCOPE
881
882
883 002326 104001
884 002330 012777 000002 176544 MOV #2,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
885 002336 005067 014114 CLR IMAX ;DON'T ITERATE TEST
886 002342 012700 000566 MOV #MPRO,R0 ;LOAD POINTER
887 002346 012703 000770 MOV #INDCO,R3 ;POINTER TO INDICATOR
888 002352 032710 000001 1$: BIT #1,@R0 ;IS THIS PARITY REGISTER PRESENT?
889 002356 001012 BNE 5$ ;NO BRANCH
890 002360 022713 000001 CMP #1,(R3) ;IS THIS CORE OR MOS PARITY REGISTER?
891 002364 001404 BEQ 6$ ;NO- BRANCH
892 002366 012770 100015 000000 MOV #100015,@(R0) ;MOS-SET ALL DEFINED BITS TO 1
893 002374 000403 BR .+10
894 002376 012770 107745 000000 6$: MOV #107745,@(R0) ;CORE- SET ALL DEFINED BITS TO 1
895 002404 062700 000010 5$: ADD #10,R0 ;MOVE POINTER TO POINT TO NEXT MPR ADDRESS
896 002410 005723 TST (R3)+ ;INCREMENT POINTER TO INDICATOR
897 ;OF A PARITY REGISTER
898 002412 020027 000766 CMP R0,#TREG ;AT END OF TABLE?
899
900 002416 002755 BLT 1$ ;NO- CONTINUE
901 002420 105767 176466 TSTB $TPFLG ;YES- TERMINAL AVAILABLE?
902 002424 001003 BNE 4$ ;NO- BRANCH
903 002426 105777 176452 TSTB @TPS ;YES- WAIT FOR TERMINAL TO FINISH
904 002432 100375 BPL .-4
905 002434 000005 4$: RESET ;ISSUE INIT
906 002436 012700 000566 MOV #MPRO,R0 ;LOAD ADDRESS OF THE TABLE
907 002442 012703 000770 MOV #INDCO,R3 ;POINTER TO INDICATOR
908 002446 105767 176550 TSTB CACHFL ;CACHE ?
909 002452 001403 BEQ 2$ ;BRANCH IF NO
910 002454 012777 000014 176542 MOV #14,@CACHE ;DISABLE CACHE
911 002462 032710 000001 2$: BIT #1,@R0 ;IS THIS PARITY REGISTER PRESENT?
912 002466 001030 BNE 3$ ;NO- BRANCH
913 002470 022713 000001 CMP #1,(R3) ;IS THIS A CORE PAR REGISTER?
914 002474 001012 BNE 7$ ;NO, BRANCH
915 002476 017002 000000 MOV @(R0),R2 ;YES, GET CONTENTS OF REGISTER
916 002502 005070 000000 CLR @(R0) ;MAKE SURE THAT WWP AND AE ARE CLEAR
917 002506 042702 077772 BIC #77772,R2 ;MASK RESERVED BITS FOR CORE PAR
918 ;-ITY REGISTER, BITS 5-11(ADDRS
919 ;BITS) ARE ALSO MASKED
920 002512 005702 TST R2 ;CHECK, IF REST WERE CLEARED
921 002514 001401 BEQ .-4
922 002516 104002 ERROR ;CORE PARITY REGISTER WHOSE ADDRESS IS
923 ;POINTED TO BY R0 WAS INCORRECT
924 ;AFTER A RESET WAS ISSUED- CONTENTS
925 ;SAVED IN R2 WITH UNUSED BITS MASKED
926
927 002520 000411 BR 3$-4
928 002522 017002 000000 7$: MOV @(R0),R2 ;MOS, GET CONTENTS OF REGISTER
929 002526 005070 000000 CLR @(R0) ;MAKE SURE THAT WWP AND AE ARE CLEAR
930 002532 046702 176276 BIC RESVM,R2 ;MASK RESERVED BITS FOR MOS PAR REG
931 002536 005702 TST R2 ;CHECK REST
932 002540 001401 BEQ .-4 ;RESET DID CLEAR ALL BITS
933 002542 104002 ERROR ;MOS PARITY REGISTER WHOSE ADDRESS IS
934 ;POINTED BY R0 WAS INCORRECT. AFTER
935 ;ISSUING RESET CONTENTS OF PAR REG
936 ;WERE AS -DOWN IN R2(UNUSED BITS
;HAVE BEEN MASKED)
    
```

```

937 002544 005070 000000 CLR @(R0) ;REINITIALIZE PARITY REGISTER
938 002550 062700 000010 3$: ADD #10,R0 ;MOVE POINTER TO POINT TO ADDRESS
939 ;OF NEXT REGISTER
940 002554 005723 TST (R3)+ ;INCREMENT POINTER TO INDICATOR
941 002556 020027 000766 CMP R0,#TREG ;DONE?
942 002562 002737 BLT 2$ ;NO- LOOP
943
944
945 ;*****
946 ;MAP CORRESPONDENCE BETWEEN PARITY REGISTERS AND MEMORY, AND TYPE RESULTS
947 ;NOTE THAT IF PARITY MEMORY IS NOT LOCATED CORRECTLY BY THIS SUBTEST
948 ;IT IS DUE TO ONE OF THE FOLLOWING FAILURES:
949 ; -SETTING WRITE WRONG PARITY DID NOT CAUSE BAD PARITY TO BE WRITTEN
950 ; -PARITY GENERATE OR DETECT LOGIC FAILED
951 ; -PARITY ERROR BIT FAILED TO SET
952 ; -PARITY BITS IN MEMORY LOCATION FAILED (I.E. BIT STUCK AT GOOD PARITY VALUE)
953 ;NOTE THAT SETTING SWITCH REGISTER SWITCH 9 WILL CAUSE A HALT AFTER THE MAP
954 ;IS TYPED. IF YOU WISH TO CHANGE THE MAP TO ISOLATE THE CAUSE OF A MAPPING
955 ;FAILURE, YOU CAN DO THIS ONCE THE PROCESSOR IS HALTED. SEE THE DESCRIPTION
956 ;IN THE LISTING (PRECEDING THE MAP TAG "MPRO" AT LOCATION 600) FOR THE MEANING
957 ;OF THE MAP CONTENTS. AFTER MAKING THE DESIRED CHANGES, PRESS CONTINUE. THE NEW
958 ;MAP WILL BE TYPED AND IF SWITCH 9 IS LEFT SET THE PROCESS WILL BE REPEATED.
959 ;IF SWITCH 9 IS NOT LEFT SET THE PROGRAM WILL PROCEED TO TEST THE PARITY MEMORY
960 ;AND REGISTERS AS RECORDED IN THE NEW MAP.
961 ;*****
962 002564 104001 TEST1: SCOPE
963 002566 012777 000003 176306 MOV #3,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
964 002574 005767 175762 TST MTYFG ;IF MAPPING HAS ALREADY BEEN DONE
965 002600 001044 BNE TEST4 ;SKIP SUBTEST
966 002602 004767 011204 JSR %7,CLRPAR ;MAP MEMORY
967 002606 004767 007144 JSR %7,MAPMEM ;FIND PARITY MEMORY AND CORRESPONDING
968 002612 004767 007466 JSR %7,MAPREG ;REGISTERS USING WRITE WRONG PARITY
969 ;WITHOUT ACTION ENABLE SET
970 ;INITIALIZE LOCATIONS INDICATING
971 002616 005067 176246 CONT3: CLR PMEML ;TOTAL PARITY MEMORY PRESENT
972 002622 005067 176244 CLR PMEMH
973 002626 012701 000566 MOV #MPRO,R1
974 002632 032711 000001 1$: BIT #1,@R1
975 002636 001006 BNE 2$
976 002640 056167 000002 176222 BIS 2(R1),PMEML ;FLAG EXISTING PARITY MEMORY (LOW 64K)
977 002646 056167 000004 176216 BIS 4(R1),PMEMH ;FLAG EXISTING PARITY MEMORY (HIGH 64K)
978 002654 062701 000010 2$: ADD #10,R1
979 002660 020127 000766 CMP R1,#TREG
980 002664 103762 BLO 1$
981 002666 004767 010126 JSR %7,TMAP ;TYPE MAP
982 002672 005267 175664 INC MTYFG ;INDICATE MAPPING DONE
983 002676 032777 001000 176174 BIT #BIT9,@SWR ;SWITCH 9 SET?
984 002704 001402 BEQ .+6 ;NO- BRANCH
985 002706 000000 HALT ;YES- SWITCH 9 SET INDICATING HALT
986 ;AFTER TYPING PARITY MEMORY MAP
987 002710 000742 BR CONT3 ;GO TYPE NEW MAP TO VERIFY USER'S INTENT
988
989
990 ;*****
991 ;SHOW THAT ASSERT PB WORKS CORRECTLY FOR EACH REGISTER
992 ;SHOW THAT NO TRAP OCCURS IF ACTION ENABLE (AE) IS NOT SET
    
```

```

993 ;SHOW THAT SETTING AE WITH ERROR ALREADY SET DOESN'T CAUSE A TRAP
994 ;NOTE THAT IF A KT11 IS PRESENT, IT IS USED DURING THIS SUBTEST
995 ;*****
996 002712 104001 TEST4: SCOPE
997 002714 012777 000004 176160 MOV #4,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
998 002722 012767 000100 013526 MOV #100,IMAX ;CLEAR ALL PARITY REGISTERS
999 002730 004767 011056 JSR %7,CLRPAR ;KT11 PRESENT?
1000 002734 005767 175610 TST NOKT ;NO-BRANCH
1001 002740 001004 BNE 1$ ;YES-MAP ALL PAGES NON RESIDENT
1002 002742 004767 010744 JSR %7,NRALL ;THEN MAP KERNEL 0 TO BANK 0, KERNEL
1003 002746 004767 011110 JSR %7,MAP1 ;7 TO EXTERNAL BANK, SET KERNEL
1004 ;0,1, AND 7 RW, AND TURN ON KT11
1005 ;SETUP TO FIND REGISTERS PRESENT
1006 002752 012700 000566 1$: MOV #MPRO,R0 ;IS THIS REGISTER PRESENT?
1007 002756 012702 000770 MOV #INDC0,R2 ;YES-BRANCH TO TEST IT
1008 002762 032710 000001 LOOP4: BIT #1,@R0 ;NO-CHECK FOR ANOTHER ONE
1009 002766 001405 BEQ TST4 ;INCREMENT PTR9LT
1010 002770 062700 000010 LOP4: ADD #10,R0 ;BRANCH WHEN ALL REGISTERS HAVE BEEN TESTED
1011 002774 005722 TST (R2)+ ;LOCATE MEMORY CORRESPONDING TO
1012 002776 103771 BLD LOOP4 ;THIS REGISTER- IF NO KT11, R1 SHOULD
1013 003000 000457 BR DONE4 ;BE RETURNED CONTAINING THE ADDRESS
1014 003002 004767 011120 TST4: JSR %7,LOCATM ;OF THE 1ST LOCATION CONTROLLED BY THIS
1015 ;REGISTER (INCLUDING EXTERNAL INTERLEAVE
1016 ;OFFSET IF NEEDED)
1017 ;IF KT11 IS PRESENT, R1 SHOULD BE RETURNED
1018 ;POINTING TO THE 1ST LOCATION CONTROLLED
1019 ;BY THIS REGISTER, MAPPED THRU KERNEL
1020 ;PAGE 1. KERNEL PAGE 1 SHOULD BE
1021 ;MAPPED TO THE CORRECT BANK
1022 ;IS ERROR RETURN INDICATED?
1023 ;NO- BRANCH
1024 003006 032701 000001 BIT #1,R1 ;MAP INDICATES NO PARITY MEMORY
1025 003012 001403 BEQ .+10 ;IS CONTROLLED BY THIS REGISTER
1026 003014 104002 ERROR ;R0 POINTS TO THE ADDRESS OF THE
1027 ;PARITY REGISTER
1028
1029
1030
1031 003016 000167 177746 JMP LOP4 ;SETUP PARITY TRAP RETURN
1032 003022 012737 003110 000114 MOV #TRP4A,@#114 ;SET WRITE WRONG PARITY
1033 003030 012770 000004 000000 MOV #NWP,@(R0) ;WRITE CONTENTS OF LOCATION WITH
1034 003036 011111 MOV @R1,@R1 ;WRONG PARITY
1035 ;CLEAR PARITY REGISTER
1036 003040 005070 000000 CLR @(R0) ;READ BAD PARITY WITH ACTION ENABLE
1037 003044 005711 TST @R1 ;CLEARED- NO TRAP EXPECTED
1038 ;CHANGE PARITY TRAP RETURN
1039 003046 012737 003116 000114 MOV #TRP4B,@#PARVEC ;SET ACTION ENABLE WITH PARITY ERROR
1040 003054 052770 000001 000000 BIS #AE,@(R0) ;ALREADY SET- SHOULDN'T TRAP YET
1041 ;CHANGE PARITY TRAP RETURN
1042 003062 012737 003124 000114 MOV #TRP4C,@#PARVEC ;READ LOCATION AGAIN- SHOULD GET
1043 003070 005711 TST @R1 ;A PARITY TRAP DUE TO READING BAD
1044 ;PARITY WITH ACTION ENABLE SET
1045 ;NO PARITY TRAP AFTER READING LOCATION
1046 003072 104002 ERROR ;WHICH SHOULD CONTAIN BAD PARITY-
1047 ;R1 CONTAINS ADDRESS OF MEMORY LOCATION
1048

```

```

1049 ;(VIRTUAL, THRU KERNEL PAGE 1, IF KT11
1050 ;PRESENT). R0 POINTS TO THE ADDRESS OF
1051 ;THE PARITY REGISTER IN WHICH AE WAS SET
1052 003074 005070 000000 CONT4: CLR @(R0) ;CLEAR PARITY REGISTER
1053 003100 005511 ADC @R1 ;CLEAR BAD PARITY
1054 003102 005070 000000 CLR @(R0) ;CLEAR PARITY ERROR BIT
1055 003106 000730 BR LOP4 ;GO TO TEST NEXT REGISTER
1056 003110 104002 TRP4A: ERROR ;PARITY TRAP OCCURRED WITH ACTION
1057 ;ENABLE CLEAR- R0 POINTS TO THE ADDRESS
1058 ;OF THE PARITY REGISTER UNDER TEST
1059 ;R1 CONTAINS THE ADDRESS OF THE MEMORY
1060 ;UNDER TEST (VIRTUAL IF KT11 IS PRESENT)
1061 ;RESTORE STACK POINTER
1061 003112 022626 CMP (SP)+,(SP)+
1062 003114 000767 BR CONT4 ;PARITY TRAP OCCURRED WHEN ACTION
1063 003116 104002 TRP4B: ERROR ;ENABLE WAS SET WITH PARITY ERROR
1064 ;ALREADY SET
1065 ;R0 POINTS TO THE ADDRESS OF THE
1066 ;PARITY REGISTER UNDER TEST
1067 ;R1 CONTAINS THE MEMORY ADDRESS UNDER
1068 ;TEST (VIRTUAL IF KT11 IS PRESENT)
1069 ;RESTORE STACK POINTER
1070 003120 022626 CMP (SP)+,(SP)+
1071 003122 000764 BR CONT4 ;PARITY TRAP OCCURRED WHEN ACTION
1072 003124 005770 000000 TRP4C: TST @(R0) ;ENABLE WAS SET WITH PARITY ERROR
1073 003130 104001 BMI .+4 ;ALREADY SET
1074 003132 104002 ERROR ;R0 POINTS TO THE ADDRESS
1075 ;OF THE PARITY REGISTER UNDER TEST
1076 ;RESTORE STACK POINTER
1077 003134 022626 CMP (SP)+,(SP)+
1078 003136 000756 BR CONT4 ;PARITY TRAP OCCURRED WHEN ACTION
1079 003140 012737 000116 000114 DONE4: MOV #PARVEC+2,@#PARVEC ;RESTORE TRAP CATCHER
1080 003146 005767 175376 TST NOKT
1081 003152 001002 BNE .+6
1082 003154 005037 177572 CLR @#SRO ;TURN OFF KT11 IF PRESENT
1083
1084
1085 ;*****
1086 ;SHOW THAT READING GOOD PARITY AFTER BAD PARITY DOESN'T CLEAR PARITY ERROR BIT
1087 ;*****
1088 003160 104001 TEST5: SCOPE
1089 003162 012777 000005 175712 MOV #5,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
1090 003170 004767 010616 JSR PC,CLRPAR ;CLEAR ALL PARITY REGISTERS
1091 003174 005767 175350 TST NOKT ;KT11 PRESENT?
1092 003200 001004 BNE 1$ ;NO, BRANCH
1093 003202 004767 010504 JSR PC,NRALL ;YES, MAP ALL PAGES NON-RESIDENT
1094 003206 004767 010650 JSR PC,MAP1 ;THEN MAP KERNEL 0 TO BANK 0. MAP KERNEL
1095 ;7 TO THE EXTERNAL BANK. SET KERNEL
1096 ;0,1, AND 7 RW, AND TURN ON KT11
1097 003212 012700 000566 1$: MOV #MPRO,R0 ;SETUP TO FIND REGISTERS PRESENT
1098 003216 032710 000001 LOOP5: BIT #1,@R0 ;IS THIS REGISTER PRESENT?
1099 003222 001406 BEQ TST5 ;YES- BRANCH TO TEST IT
1100 003224 062700 000010 LOP5: ADD #10,R0 ;NO- CHECK FOR ANOTHER ONE
1101 003230 020027 000766 CMP RO,#TREG
1102 003234 103770 BLD LOOP5
1103 003236 000431 BR DONE5 ;EXIT WHEN ALL REGISTERS HAVE BEEN TESTED
1104 003240 004767 010662 TST5: JSR PC,LOCATM ;LOCATE MEMORY CORRESPONDING TO THIS

```

```

1105 ;REGISTER- R1 WILL BE RETURNED CONTAINING
1106 ;THE ADDRESS OF THE FIRST LOCATION
1107 ;CONTROLLED BY THIS REGISTER (MAPPED
1108 ;THRU KERNEL PAGE 1 IF KT11 IS PRESENT)
1109 003244 032701 000001 BIT #1,R1 ;IS ERROR RETURN INDICATED?
1110 003250 001403 BEQ .+10 ;NO- BRANCH
1111 003252 104002 ERROR ;MAP INDICATES NO PARITY MEMORY IS
1112 ;CONTROLLED BY THIS REGISTER. R0
1113 ;POINTS TO THE ADDRESS OF THE
1114 ;PARITY REGISTER
1115 003254 000167 177744 JMP LOP5 ;SET WWP IN THIS REGISTER
1116 003260 012770 000004 000000 MOV #WWP,@(R0) ;WRITE CONTENTS OF LOCATION WITH
1117 003266 011111 MOV @R1,@R1 ;WRONG PARITY
1118 ;DETECT WRONG PARITY
1119 003270 005711 TST @R1 ;CLEAR WWP IN PARITY REGISTER
1120 003272 042770 000004 000000 BIC #WWP,@(R0) ;RESTORE GOOD PARITY
1121 003300 011111 MOV @R1,@R1 ;REREAD LOCATION
1122 003302 005711 TST @R1 ;READ CONTENTS OF PARITY REGISTER
1123 003304 005770 000000 TST @(R0) ;BRANCH IF PARITY ERROR IS STILL SET
1124 003310 100401 BMI .+4 ;PARITY ERROR BIT CLEARED BY READING
1125 003312 104002 ERROR ;GOOD PARITY (OR POSSIBLY WHILE DOING
1126 ;THE BIC TO CLEAR WWP). R0 POINTS TO
1127 ;THE PARITY REGISTER ADDRESS.
1128 ;CLEAR THE PARITY REGISTER
1129 003314 005070 000000 CLR @(R0)
1130 003320 000741 BR LOP5
1131 003322 005767 175222 DONE5: TST NOKT
1132 003326 001002 BNE .+6
1133 003330 005037 177572 CLR @#SRO ;TURN OFF KT11 IF PRESENT
1134
1135
1136 ;*****
1137 ;SHOW THAT PARITY GENERATE AND DETECT LOGIC WORKS CORRECTLY FOR EACH BYTE
1138 ;SHOW THAT WRITE WRONG PARITY WORKS FOR HIGH AND LOW BYTES
1139 ;SHOW THAT WRITING INTO LOCATION WHEN WRITE WRONG PARITY IS NOT SET
1140 ;RESTORES GOOD PARITY
1141 ;*****
1142 003334 104001 TEST6: SCOPE
1143 003336 012777 000006 175536 MOV #6,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
1144 003344 004767 010442 JSR %7,CLRPAR ;CLEAR ALL PARITY REGISTERS
1145 003350 005767 175174 TST NOKT ;KT11 PRESENT?
1146 003354 001004 BNE 1$ ;NO, BRANCH
1147 003356 004767 010330 JSR %7,NRALL ;YES- MAP IT (KERNEL 0 TO BANK 0, RW;
1148 003362 004767 010474 JSR %7,MAP1 ;KERNEL 7 TO EXTERNAL BANK, RW; KERNEL 1 RW)
1149 ;AND TURN IT ON
1150 003366 012700 000566 1$: MOV #MPRO,R0 ;SETUP TO FIND REGISTERS PRESENT
1151 003372 032710 000001 LOOP6: BIT #1,@R0 ;IS THIS REGISTER PRESENT?
1152 003376 001406 BEQ 2$ ;YES- BRANCH TO TEST IT
1153 003400 062700 000010 LOP6: ADD #10,R0 ;NO- CHECK FOR ANOTHER ONE
1154 003404 020027 000766 CMP R0,#TREG
1155 003410 103770 BLO LOP6
1156 003412 000523 BR DONE6 ;BRANCH TO DONE IF ALL REGISTERS
1157 ;HAVE BEEN TESTED
1158 003414 004767 010506 TST6: JSR %7,LOCATM ;LOCATE MEMORY CORRESPONDING TO
1159 ;THIS REGISTER- R1 SHOULD BE RETURNED
1160 ;CONTAINING THE ADDRESS OF THE FIRST
    
```

```

1161 ;LOCATION CONTROLLED BY THIS REGISTER
1162 ;(VIRTUAL THRU KERNEL PAGE 1 IF KT11 PRESENT)
1163 003420 032701 000001 BIT #1,R1 ;IF NO CORRESPONDING
1164 003424 001403 BEQ .+10 ;ADDRESS WAS FOUND TO CORRESPOND
1165 003426 104002 ERROR ;ODD ADDRESS IS RETURNED-BRANCH IF OK
1166 ;MAP INDICATES NO PARITY MEMORY IS
1167 ;CONTROLLED BY THIS REGISTER
1168 ;R0 POINTS TO THE ADDRESS OF THE
1169 003430 000167 177744 JMP LOP6 ;PARITY REGISTER
1170 ;AFTER ERROR, CHECK FOR NEXT REGISTER
1171 ;FIRST SHOW THAT IF THE PARITY REGISTER IS CLEARED INITIALLY,
1172 ;PARITY ERROR DOESN'T SET
1173 003434 005011 CLR @R1 ;INITIALIZE LOCATION UNDER TEST
1174 003436 005070 000000 CLR @(R0) ;INITIALLY CLEAR PARITY REGISTER
1175 003442 005002 CLR R2 ;R2 CONTAINS VALUE TO BE LOADED
1176 ;INTO MEMORY
1177 003444 110211 1$: MOVB R2,@R1 ;WRITE VALUE INTO LOW BYTE
1178 003446 005711 TST @R1 ;READ WORD TO CHECK PARITY
1179 003450 005770 000000 TST @(R0) ;CHECK PARITY REGISTER
1180 003454 100006 BPL 5$ ;BRANCH IF ERROR NOT SET
1181 003456 104002 ERROR ;PARITY ERROR SET WHEN VALUE IN R2 WAS
1182 ;WRITTEN AND READ BACK FROM LOW BYTE OF
1183 ;LOCATION WHOSE ADDRESS IS CONTAINED IN
1184 ;R1 (WWP WAS NOT SET)
1185 003460 005070 000000 CLR @(R0) ;CLEAR ERROR BIT
1186 003464 005011 CLR @R1 ;REINITIALIZE TEST LOCATION
1187 003466 005002 CLR R2 ;REINITIALIZE VALUE TO BE USED
1188 003470 000402 BR 2$
1189 003472 105202 5$: INCB R2 ;INCREMENT VALUE TO BE LOADED
1190 003474 001363 BNE 1$ ;LOOP UNTIL ALL VALUES HAVE BEEN USED
1191 003476 110261 2$: MOVB R2,1(R1) ;WRITE ALL VALUES INTO HIGH BYTE
1192 003502 005711 TST @R1 ;READ WORD TO CHECK PARITY
1193 003504 005770 000000 TST @(R0) ;CHECK PARITY REGISTER
1194 003510 100002 BPL .+6 ;BRANCH IF ERROR NOT SET
1195 003512 104002 ERROR ;PARITY ERROR SET WHEN VALUE IN R2
1196 ;WAS WRITTEN AND READ BACK FROM HIGH BYTE
1197 ;OF LOCATION WHOSE ADDRESS IS CONTAINED
1198 003514 000402 BR 6$ ;IN R1 (WWP WAS NOT SET)
1199 003516 105202 INCB R2 ;INCREMENT VALUE TO BE LOADED
1200 003520 001366 BNE 2$ ;LOOP UNTIL ALL VALUES HAVE BEEN USED
1201
1202 ;TEST PARITY GENERATE AND DETECT LOGIC BY SETTING WRITE WRONG PARITY AND
1203 ;WRITING EACH POSSIBLE VALUE TO THE LOW BYTE, THEN TO THE HIGH BYTE
1204 003522 005011 6$: CLR @R1 ;INITIALIZE LOCATION UNDER TEST
1205 003524 005002 CLR R2 ;INITIALIZE VALUE TO BE WRITTEN
1206 003526 012770 000004 000000 3$: MOV #WWP,@(R0) ;SET WRITE WRONG PARITY
1207 003534 110211 MOVB R2,@R1 ;WRITE WRONG PARITY IN LOW BYTE
1208 003536 005070 000000 CLR @(R0) ;CLEAR WRITE WRONG PARITY, AND CLEAR
1209 ;PARITY ERROR IF SET
1210 003542 005711 TST @R1 ;READ BACK WRONG PARITY
1211 003544 005770 000000 TST @(R0) ;PARITY ERROR SET?
1212 003550 100402 BMI .+6 ;YES-BRANCH
1213 003552 104002 ERROR ;PARITY ERROR DID NOT SET WHEN THE
1214 ;LOCATION UNDER TEST WAS WRITTEN
1215 ;AND READ BACK WITH WRITE WRONG PARITY
1216 ;SET. R0 POINTS TO ADDRESS OF PARITY
    
```

```

1217 ;REGISTER, R1 CONTAINS ADDRESS OF LOCATION
1218 ;BEING TESTED (VIRTUAL THRU KERNEL
1219 ;PAGE 1 IF KT11 IS PRESENT). R2
1220 ;CONTAINS THE VALUE WRITTEN
1221 003554 000402 BR .+6 ;EXIT LOOP AFTER ERROR
1222 003556 105202 INCB R2 ;INCREMENT DATA
1223 003560 001362 BNE 3$ ;LOOP TILL DONE WITH ALL VALUES
1224 003562 005011 CLR @R1 ;REINITIALIZE LOCATION TO CLEAR BAD PARITY
1225 003564 005070 000000 CLR @(R0) ;CLEAR ERROR IF SET
1226 003570 005711 TST @R1 ;READ LOCATION, WHICH SHOULD NOW HAVE
1227 ;GOOD PARITY
1228 003572 005770 000000 TST @(R0) ;PARITY ERROR SET?
1229 003576 100301 BPL .+4 ;NO, BRANCH
1230 003600 104002 ERROR ;GOOD PARITY WAS NOT RESTORED BY
1231 ;WRITING INTO THE LOCATION WITH
1232 ;WRITE WRONG PARITY CLEARED
1233 003602 012770 000004 000000 4$: MOV #WVP,@(R0) ;SET WRITE WRONG PARITY
1234 003610 110261 000001 MOVB R2,1(R1) ;WRITE WRONG PARITY IN HIGH BYTE
1235 003614 005070 000000 CLR @(R0) ;CLEAR WRITE WRONG PARITY AND PARITY
1236 ;ERROR IF SET
1237 003620 005711 TST @R1 ;READ BACK WRONG PARITY
1238 003622 005770 000000 TST @(R0) ;PARITY ERROR SET?
1239 003626 100402 BMI .+6 ;YES-BRANCH
1240 003630 104002 ERROR ;PARITY ERROR DID NOT SET WHEN THE LOCATION
1241 ;UNDER TEST WAS WRITTEN AND READ BACK WITH
1242 ;WRITE WRONG PARITY SET. R0 POINTS
1243 ;TO THE ADDRESS OF THE PARITY REGISTER.
1244 ;THE VALUE IN R2 WAS WRITTEN INTO THE HIGH
1245 ;BYTE OF THE LOCATION WHOSE ADDRESS IS IN R1
1246 ;(VIRTUAL THRU KERNEL PAGE 1 IF KT11 PRESENT)
1247 ;THEN THE PARITY REGISTER WAS
1248 ;CLEARED AND THE WORD WAS READ BACK
1249 003632 000402 BR .+6 ;EXIT LOOP AFTER ERROR
1250 003634 105202 INCB R2 ;INCREMENT DATA
1251 003636 001361 BNE 4$ ;LOOP TILL DONE WITH ALL VALUES
1252 003640 005011 CLR @R1 ;CLEAR BAD PARITY IN TEST LOCATION
1253 003642 005070 000000 CLR @(R0) ;CLEAR PARITY ERROR BIT IF SET
1254 003646 005711 TST @R1 ;READ LOCATION, WHICH SHOULD NOW
1255 ;HAVE GOOD PARITY
1256 003650 005770 000000 TST @(R0) ;CHECK PARITY ERROR BIT
1257 003654 100001 BPL .+4 ;BRANCH IF CLEAR
1258 003656 104002 ERROR ;WRITING INTO LOCATION WHEN WRITE
1259 ;WRONG PARITY WAS NOT SET DID NOT
1260 ;WRITE GOOD PARITY
1261 003660 000647 BR LOP6 ;GO CHECK FOR ANOTHER PARITY REGISTER
1262 003662 005767 174662 DONE: TST NOKT
1263 003666 001002 BNE .+6
1264 003670 005037 177572 CLR @#SR0 ;TURN OFF KT11 IF PRESENT
1265
1266
1267
1268 ;*****
1269 ;SHOW THAT SETTING PARITY ERROR AFTER SETTING ACTION ENABLE WON'T CAUSE A TRAP
1270 ;*****
1271 003674 104001 TEST7: SCOPE
1272 003676 012777 000007 175176 MOV #7,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
    
```

```

1273 003704 004767 010102 JSR %7,CLRPAR ;INITIALLY CLEAR ALL PARITY REGISTERS
1274 003710 012737 004004 000114 MOV #TRP7,@#PARVEC ;SETUP PARITY TRAP RETURN
1275 003716 005037 000116 CLR @#PARVEC+2
1276 003722 012700 000566 MOV #MPRO,R0 ;SETUP TO GET ADDRESS OF REGISTER PRESENT
1277 003726 032710 000001 LUP7: BIT #1,@R0
1278 003732 001406 BEQ TST7 ;BRANCH TO TEST REGISTER
1279 003734 062700 000010 LOOP7: ADD #10,R0
1280 003740 020027 000766 CMP R0,#TREG
1281 003744 103770 BLO LUP7
1282 003746 000412 BR DONE7 ;BRANCH IF DONE TESTING ALL REGISTERS
1283 003750 012770 000001 000000 TST7: MOV #AE,@(R0) ;SET ACTION ENABLE
1284 003756 052770 100000 000000 BIS #PERR,@(R0) ;SET PARITY ERROR
1285 003764 000240 NOP ;SHOULD NOT TRAP
1286 003766 005070 000000 CLR @(R0) ;CLEAR PARITY REGISTER
1287 003772 000760 BR LOOP7 ;GO CHECK NEXT REGISTER
1288 003774 012737 000116 000114 DONE7: MOV #PARVEC+2,@#PARVEC
1289 004002 000405 BR TEST10
1290 004004 104002 TRP7: ERROR ;TRAP OCCURRED WHEN PARITY ERROR BIT
1291 ;WAS SET VIA A BIS INSTRUCTION
1292 ;WITH ACTION ENABLE ALREADY SET.
1293 ;R0 POINTS TO THE ADDRESS OF THE
1294 ;PARITY REGISTER
1295 004006 022626 CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
1296 004010 005070 000000 CLR @(R0) ;CLEAR PARITY REGISTER
1297 004014 000747 BR LOOP7
1298
1299
1300 ;*****
1301 ;SHOW THAT REPEATED PARITY ERRORS WILL CAUSE REPEATED TRAPS IF ACTION
1302 ;ENABLE IS SET AND PARITY ERROR IS LEFT SET.
1303 ;SHOW THAT THE ERROR ADDRESS BITS (11-5) TRACK (ONLY FOR CORE PARITY REGISTERS)
1304 ;*****
1305
    
```

```
1306 004016 104001          TEST10: SCOPE
1307 004020 012777 000010 175054  MOV    #10,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
1308 004026 004767 007760  JSR    %7,CLRPAR ;INITIALLY CLEAR ALL PARITY REGISTERS
1309 004032 005767 174512  TST   NOKT ;KT11 PRESENT?
1310 004036 001004          BNE   $ ;NO- BRANCH
1311 004040 004767 007646  JSR    %7,NRALL ;YES- INITIALLY MAP ALL PAGES NR
1312 004044 004767 010012  JSR    %7,MAP1 ;MAP KERNEL 0 TO BANK 0,RW
1313                                     ;KERNEL 7 TO THE EXTERNAL BANK, RW
1314                                     ;MAKE KERNE'. PAGE 1 RW AND TURN ON THE KT11
1315 004050 012700 000566 1$:    MOV    #MPRO,R0
1316 004054 012702 000770          MOV    #INDCO,R2
1317 004060 032710 000001  LUP10: BIT  #1,@R0
1318 004064 001407          BEQ   TST10 ;BRANCH TO TEST REGISTER IF PRESENT
1319 004066 062700 000010  LOOP10: ADD #10,R0
1320 004072 005722          TST   (R2)+
1321 004074 020027 000766          CMP   RO,#TREG
1322 004100 103767          BLO  LUP10 ;BRANCH IF ALL REGISTERS HAVE BEEN TESTED
1323 004102 000507          BR   DONE10
1324 004104 004767 010016  TST10: JSR   %7,LOCATM
1325 004110 032701 000001          BIT  #1,R1 ;ERROR RETURN INDICATED?
1326 004114 001403          BEQ  .+10 ;BRANCH IF NO
1327 004116 104002          ERROR ;MAP INDICATES THERE IS NO MEMORY
1328                                     ;CORRESPONDING TO THIS REGISTER
1329                                     ;RO POINTS TO THE ADDRESS OF
1330                                     ;THE PARITY REGISTER
1331 004120 000167 177742          JMP   LOOP10
1332 004124 012770 000004 000000  MOV    #WWP,@(R0) ;SET WRITE WRONG PARITY
1333 004132 011111          MOV    @R1,@R1 ;WRITE WRONG PARITY IN FIRST LOCATION
1334 004134 016161 004000 004000  MOV    4000(R1),4000(R1) ;WRITE WRONG PARITY IN SECOND LOCATION
1335 004142 012770 000001 000000  MOV    #AE,@(R0) ;SET ACTION ENABLE AND CLEAR REST
1336 004150 012737 004202 000114  MOV    #TRP10,@#PARVEC ;SETUP PARITY TRAP RETURN
1337 004156 012767 000010 000152  MOV    #10,COUNT ;SETUP COUNTER TO EXECUTE INSTRUCTION 1
1338                                     ;(INST1) TEN TIMES
1339 004164 005711          INST1: TST  @R1 ;READ WRONG PARITY WITH AE SET- SHOULD
1340                                     ;TRAP TO TRP10
1341 004166 104002          ERROR ;NO PARITY TRAP OCCURRED. RO POINTS TO
1342                                     ;ADDRESS OF THE PARITY REGISTER BEING
1343                                     ;TESTED.
1344 004170 000441          INST2: BR   CONT10
1345 004172 005761 004000          TST  4000(R1) ;READ WRONG PARITY FROM SECOND ADDRESS
1346                                     ;WITH AE SET- SHOULD TRAP TO TRP10A
1347 004176 104002          ERROR ;NO PARITY TRAP OCCURRED. RO POINTS TO
1348                                     ;THE ADDRESS OF THE PARITY REGISTER
1349                                     ;BEING TESTED
1350 004200 000435          TRP10: BR   CONT10
1351 004202 005367 000130          DEC  COUNT ;HAS PARITY TRAP OCCURRED TEN TIMES?
1352 004206 001413          BEQ   $ ;YES- BRANCH
1353 004210 022712 000001          CMP   #1,(R2) ;IS THIS A CORE PAR REG?
1354 004214 001005          BNE  2$ ;NO, BRANCH (NO ERROR
1355                                     ;ADDRESS BITS FOR MOS PAR
1356 004216 032770 000040 000000  BIT   #BITS,@(R0) ;IF ERROR ADDRESS BITS ARE TRACKING,
1357                                     ;BIT 5 SHOULD BE CLEAR (ONLY FOR CORE PARITY)
1358 004224 001401          BEQ  .+4 ;PARITY ERROR ADDRESS BITS INCORRECT
1359 004226 104002          ERROR ;RO POINTS TO THE ADDRESS OF THE PARITY
1360                                     ;REGISTER. R1 CONTAINS THE ADDRESS
1361
```

```
1362                                     ;REFERENCED TO CAUSE A PARITY TRAP
1363                                     ;(VIRTUAL IF KT11 IS PRESENT)
1364 004230 012716 004164          2$:    MOV    #INST1,@SP ;GO EXECUTE INSTRUCTION 1 AGAIN
1365 004234 000002          RTI                                     ;CHANGE PARITY TRAP RETURN
1366 004236 012737 004252 000114 1$:    MOV    #TRP10A,@#PARVEC ;GO EXECUTE INSTRUCTION 2
1367 004244 012716 004172          MOV    #INST2,@SP
1368 004250 000002          RTI
1369 004252 022712 000001  TRP10A: CMP  #1,(R2) ;IS THIS A CORE REG?
1370 004256 001005          BNE  1$ ;NO, BRANCH
1371 004260 032770 000040 000000  BIT   #BITS,@(R0) ;PARITY TRAP OCCURRED- CHECK PARITY
1372                                     ;ERROR ADDRESS BITS
1373 004266 001001          BNE  .+4 ;BRANCH IF OK (IF THE PARITY ERROR
1374                                     ;ADDRESS BITS TRACKED, BIT 5 WILL BE SET)
1375 004270 104002          ERROR ;PARITY ERROR ADDRESS BITS INCORRECT
1376                                     ;RO POINTS TO THE ADDRESS OF THE
1377                                     ;PARITY REGISTER. THE ADDRESS REFERENCED
1378                                     ;TO CAUSE THE ERROR WAS THAT IN
1379                                     ;R1 PLUS 4000 (OCTAL).
1380 004272 022626          1$:    CMP   (SP)+,(SP)+
1381 004274 012737 000116 000114 CONT10: MOV  #PARVEC+2,@#PARVEC ;RESTORE TRAPCATCHER
1382 004302 005070 000000          CLR  @(R0) ;CLEAR PARITY REGISTER
1383 004306 005511          ADC  @R1 ;CLEAR BAD PARITY
1384 004310 005561 004000          ADC  4000(R1)
1385 004314 005070 000000          CLR  @(R0) ;CLEAR PARITY ERROR BIT IF SET
1386 004320 000662          BR   LOOP10
1387 004322 005767 174222  DONE10: TST  NOKT
1388 004326 001002          BNE  .+6 ;TURN OFF KT11 IF PRESENT
1389 004330 005037 177572          CLR  @#SRO
1390 004334 000401          BR   TEST11
1391 004336 000000          COUNT: 0
1392
1393
1394
1395 ;*****
1396 ;IF MULTIPLE PARITY ERRORS OCCUR DURING ONE INSTRUCTION (WITH ACTION ENABLE
1397 ;NOT SET) THE ERROR ADDRESS BITS WILL RECORD THE LAST ERROR (ONLY FOR CORE PARITY
1398 ;REGISTERS)
1399 ;*****
1400 004340 104001          TEST11: SCOPE
1401 004342 012777 000011 174532  MOV    #11,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
1402 004350 004767 007436  JSR    %7,CLRPAR ;INITIALLY CLEAR ALL PARITY REGISTERS
1403 004354 005767 174170  TST   NOKT ;KT11 PRESENT?
1404 004360 001004          BNE  $ ;NO- BRANCH
1405 004362 004767 007324  JSR    %7,NRALL ;YES, MAP KERNEL PAGE 0 TO BANK 0,RW
1406 004366 004767 007470  JSR    %7,MAP1 ;MAP KERNEL PAGE 7 TO THE EXTERNAL BANK, RW
1407                                     ;SET KERNEL PAGE 1 RW AND TURN ON KT11
1408                                     ;SETUP TO GET ADDRESSES OF REGISTERS PRESENT
1409 004372 012700 000566 1$:    MOV    #MPRO,R0
1410 004376 012703 000770          MOV    #INDCO,R3
1411 004402 032710 000001  LUP11: BIT  #1,@R0
1412 004406 001003          BNE  LOOP11 ;IF THIS REG NOT PRESENT, SKIP
1413 004410 022713 000001          CMP   #1,(R3) ;IS THIS A CORE PAR REG?
1414 004414 001407          BEQ  TST11 ;YES, THEN TEST IT
1415                                     ;IF NOT CORE, SKIP THIS REGISTER
1416 004416 062700 000010  LOOP11: ADD #10,R0
1417 004422 005723          TST  (R3)+
1418 004424 020027 000766          CMP  RO,#TREG
```


1418	004430	103764		BLO	LUP11		
1419	004432	000443		BR	DONE11		
1420	004434	004767	007466	TST11:	JSR	%7,LOCATM	;BRANCH OUT IF ALL REGISTERS HAVE BEEN TESTED
1421							;GET THE ADDRESS OF A MEMORY LOCATION
1422	004440	032701	000001		BIT	#1,R1	;CORRESPONDING TO THIS PARITY REGISTER
1423	004444	001403			BE	+.10	;ERROR RETURN INDICATED?
1424	004446	104002			ERROR		;BRANCH IF NOT
1425							;NO MEMORY IN MAP CORRESPONDING TO
1426							;THIS PARITY REGISTER. R0 POINTS
1427	004450	000167	177742		JMP	LOOP11	;TO THE ADDRESS OF THE PARITY REGISTER
1428	004454	010102			MOV	R1,R2	
1429	004456	062702	010000		ADD	#10000,R2	;SETUP SECOND TEST ADDRESS LOCATION
1430	004462	012770	000004	000000	MOV	#WWP,@(R0)	
1431	004470	011111			MOV	@R1,@R1	;SET WRITE WRONG PARITY
1432	004472	011212			MOV	@R2,@R2	;WRITE WRONG PARITY IN FIRST TEST LOCATION
1433	004474	005070	000000		CLR	@(R0)	;WRITE WRONG PARITY IN SECOND TEST LOCATION
1434	004500	021112			CMP	@R1,@R2	;CLEAR PARITY REGISTER
1435							;READ FIRST TEST LOCATION, AND
1436	004502	005770	000000		TST	@(R0)	;THEN READ SECOND TEST LOCATION
1437	004506	100401			BMI	+.4	;MAKE SURE PARITY ERROR SET
1438	004510	104002			ERROR		
1439							;PARITY ERROR NOT SET AFTER
1440							;READING TWO LOCATIONS WHICH
1441	004512	032770	000100	000000	BIT	#BIT6,@(R0)	;SHOULD HAVE BAD PARITY
1442							;CHECK ERROR ADDRESS- IF THE LAST
1443							;ADDRESS WAS RECORDED. BIT 6 WILL
1444	004520	001001			BNE	+.4	;BE SET
1445	004522	104002			ERROR		
1446							;PARITY ERROR ADDRESS BITS INCORRECT
1447							;R0 POINTS TO ADDRESS OF PARITY REGISTER
1448							;R2 CONTAINS ADDRESS OF LAST BAD PARITY
1449							;LOCATION REFERENCED (IF KT11 PRESENT,
1450	004524	005070	000000		CLR	@(R0)	;ADDRESS IS VIRTUAL THRU KERNEL PAGE 1)
1451	004530	005511			ADC	@R1	;CLEAR PARITY REGISTER
1452	004532	005512			ADC	@R2	;CLEAR BAD PARITY
1453	004534	005070	000000		CLR	@(R0)	;CLEAR PARITY ERROR BIT
1454	004540	000726			BR	LOOP11	
1455	004542	005767	174002	DONE11:	TST	NOKT	
1456	004546	001002			BNE	+.6	
1457	004550	005037	177572		CLR	@#SRO	;TURN OFF KT11 IF PRESENT
1458							
1459							
1460							
1461							
1462							
1463							
1464							
1465							
1466	004554	104001					
1467	004556	012777	000012	174316	TEST12:	SCOPE	
1468	004564	004767	007222		MOV	#12,@DISPLAY	;LOAD THE TEST NUMBER INTO THE DISPLAY
1469	004570	005767	173754		JSR	%7,CLRPAR	
1470	004574	001004			TST	NOKT	;KT11 PRESENT?
1471	004576	004767	007110		BNE	1\$;NO- BRANCH
1472	004602	004767	007254		JSR	%7,NRALL	;YES, MAP KERNEL 0 TO BANK 0, RW
1473					JSR	%7,MAP1	;MAP KERNEL 7 TO THE EXTERNAL BANK, RW
							;SET KERNEL 1 RW AND TURN ON KT11

1474	004606	012700	000566	1\$:	MOV	#MPRO,R0	;SETUP TO GET ADDRESSES OF REGISTERS PRESENT
1475	004612	012702	000770		MOV	#INDCO,R2	
1476	004616	032710	000001	LUP12:	BIT	#1,@R0	
1477	004622	001003			BNE	LOOP12	;SKIP, IF THIS REG NOT PRESENT
1478	004624	022712	000001		CMP	#1,(R2)	;REG PRESENT, IS IT CORE?
1479	004630	001407			BEQ	TST12	;YES, DO THIS TEST
1480							;SKIP, IF THIS REG IS NOT CORE
1481	004632	062700	000010	LOOP12:	ADD	#10,R0	
1482	004636	005722			TST	(R2)+	
1483	004640	020027	000766		CMP	R0,#TREG	
1484	004644	103764			BLO	LUP12	
1485	004646	000474			BR	DONE12	
1486							;BRANCH TO DONE IF ALL REGISTERS
1487	004650	004767	007252	TST12:	JSR	%7,LOCATM	;HAVE BEEN TESTED
1488							;LOCATE MEMORY CORRESPONDING TO THIS
1489	004654	032701	000001		BIT	#1,R1	;REGISTER
1490	004660	001403			BEQ	+.10	;ERROR RETURN INDICATED?
1491	004662	104002			ERROR		;NO- BRANCH
1492							;NO MEMORY IN MAP CORRESPONDING TO
1493							;THIS REGISTER. R0 POINTS TO
1494	004664	000167	177742		JMP	LOOP12	;THE ADDRESS OF THE PARITY REGISTER
1495	004670	012737	004724	000114	MOV	#TRP12,@#PARVEC	;SET UP PARITY TRAP RETURN
1496	004676	012770	000004	000000	MOV	#WWP,@(R0)	;SET WRITE WRONG PARITY
1497	004704	012711	125252		MOV	#125252,@R1	;WRITE WRONG PARITY IN TEST LOCATION
1498	004710	012770	000001	000000	MOV	#AE,@(R0)	;SET ACTION ENABLE AND CLEAR
1499							;WRITE WRONG PARITY
1500	004716	005211			INC	@R1	;DO DATIP.DATO WITH ACTION ENABLE
1501							;SET- SHOULD ABORT ON DATIP AND
1502	004720	104002			ERROR		;RESTORE ORIGINAL DATA
1503							;NO ABORT OCCURRED ON READING LOCATION
1504							;WHICH SHOULD CONTAIN BAD PARITY
1505							; (WITH AE SET).
1506							;R0 POINTS TO ADDRESS OF PARITY REGISTER.
1507							;R1 CONTAINS ADDRESS OF TEST LOCATION
1508							; (VIRTUAL THRU KERNEL PAGE 1 IF KT11 IS
1509							;PRESENT)
1510	004722	000440			BR	CONT12	
1511	004724	005070	000000	TRP12:	CLR	@(R0)	;PARITY TRAP OCCURRED- CLEAR PARITY REGISTER
1512	004730	021127	125252		CMP	@R1,#125252	;ORIGINAL DATA RESTORED?
1513	004734	001401			BEQ	+.4	;YES, BRANCH
1514	004736	104002			ERROR		;NO- DATIP WHICH GOT A PARITY ERROR
1515							;TRAP ALTERRED CONTENTS OF LOCATION
1516							;READ. ADDRESS OF TEST LOCATION IS IN R1
1517							; (IF KT11 IS PRESENT, ADDRESS IN R1
1518							; IS VIRTUAL THRU KERNEL PAGE 1)
1519							;R0 POINTS TO ADDRESS OF PARITY REGISTER
1520	004740	005770	000000		TST	@(R0)	;MAKE SURE PARITY ERROR SET WHEN
1521	004744	100401			BMI	+.4	;DATA WAS REREAD IN THE ABOVE CMP
1522	004746	104002			ERROR		;DATIP WHICH GOT A PARITY ERROR TRAP
1523							;ALTERRED THE PARITY OF THE LOCATION READ
1524							;R1 CONTAINS ADDRESS OF TEST LOCATION
1525							; (VIRTUAL THRU KERNEL 1 IF KT11 PRESENT)
1526	004750	022626			CMP	(SP)+,(SP)+	;RESTORE STACK POINTER
1527	004752	012770	000004	000000	MOV	#WWP,@(R0)	;SET WRITE WRONG PARITY AND CLEAR
1528							;PARITY ERROR
1529	004760	012711	125252		MOV	#125252,@R1	;REWRITE DATA WITH WRONG PARITY

```

1530 004764 005070 000000          CLR    @(R0)          ;CLEAR PARITY REGISTER
1531 004770 012737 000116 000114  MOV    #PARVEC+2,@#PARVEC ;RESTORE TRAPCATCHER
1532 004776 005211                INC    @R1            ;SINCE AE IS CLEAR, INSTRUCTION SHOULD
1533                                ;COMPLETE AND SHOULD CLEAR BAD PARITY
1534 005000 005070 000000          CLR    @(R0)          ;CLEAR PARITY ERROR BIT
1535 005004 022711 125253          CMP    #125253,@R1   ;CHECK DATA
1536 005010 001401                BEQ    .+4            ;DATI, DATO TO A LOCATION CONTAINING BAD
1537 005012 104002                ERROR           ;PARITY WITHOUT AE SET LEFT INCORRECT
1538                                ;DATA, R0 POINTS TO THE ADDRESS OF
1539                                ;THE PARITY REGISTER. R1 CONTAINS THE
1540                                ;ADDRESS OF THE TEST LOCATION (VIRTUAL
1541                                ;THRU KERNEL PAGE 1 IF KT11 IS PRESENT)
1542                                ;CHECK PARITY ERROR BIT
1543 005014 005770 000000          TST    @(R0)          ;DATI, DATO WITH AE CLEAR DID
1544 005020 100001                BPL    .+4            ;NOT CLEAR BAD PARITY IN LOCATION
1545 005022 104002                ERROR           ;ADDRESSED, R0 POINTS TO THE ADDRESS
1546                                ;OF THE PARITY REGISTER, R1 CONTAINS
1547                                ;THE ADDRESS OF THE TEST LOCATION
1548                                ;(VIRTUAL THRU KERNEL PAGE 1 IF KT11
1549                                ;IS PRESENT)
1550                                ;CLEAR PARITY REGISTER
1551                                ;CLEAR LOCATION TO RESTORE GOOD PARITY
1552 005024 005070 000000  CONT12: CLR    @(R0)
1553 005030 005011                CLR    @R1            ;CLEAR PARITY ERROR IF SET
1554 005032 005070 000000          CLR    @(R0)
1555 005036 000675                BR     LOOP12         ;GO CHECK FOR ANOTHER PARITY
1556                                ;REGISTER
1557 005040 012737 000116 000114  DONE12: MOV   #PARVEC+2,@#PARVEC ;RESTORE TRAPCATCHER
1558 005046 005767 173476          TST    NOKT          ;
1559 005052 001002                BNE    .+6            ;
1560 005054 005037 177572          CLR    @#SR0         ;TURN OFF KT11 IF PRESENT
1561
1562
1563
1564
1565 ;*****
1566 ;SHOW THAT IF AN INSTRUCTION DOING A DATI (BUT NO DATO TO THE SAME LOCATION)
1567 ;GETS A PARITY ERROR, THE ORIGINAL DATA IS UNALTERED, WHETHER OR NOT ACTION
1568 ;ENABLE IS SET
1569 ;*****
1570 005060 104001                TEST13: SCOPE
1571 005062 012777 000013 174012  MOV    #13,@DISPLAY   ;LOAD THE TEST NUMBER INTO THE DISPLAY
1572 005070 004767 006716          JSR    %7,CLRPAR     ;
1573 005074 005767 173450          TST    NOKT          ;KT11 PRESENT?
1574 005100 001004                BNE    1$            ;NO- BRANCH
1575 005102 004767 006604          JSR    %7,NRALL      ;YES, MAP KERNEL 0 TO BANK 0, KERNEL
1576 005106 004767 006750          JSR    %7,MAP1       ;7 TO THE EXTERNAL BANK, AND KERNEL
1577                                ;0,1,AND 7 RW
1578 005112 012700 000566          1$:  MOV    #MPRO,R0   ;SETUP TO GET ADDRESSES OF REGISTERS PRESENT
1579 005116 032710 000001  LUP13: BIT    #1,@R0
1580 005122 001406                BEQ    TST13         ;IF THIS REGISTER IS PRESENT, GO
1581                                ;TEST IT
1582 005124 062700 000010          LOOP13: ADD   #10,R0
1583 005130 020027 000766          CMP    R0,#TREG     ;
1584 005134 103770                BLD    LUP13         ;
1585 005136 000470                BR     DONE13        ;BRANCH I: ALL REGISTERS HAVE BEEN
1586                                ;TESTED

```

```

1586 005140 004767 006762          TST13: JSR    %7,LOCATM ;LOCATE MEMORY CORRESPONDING TO THIS
1587                                ;REGISTER, AND IF KT11 IS PRESENT
1588                                ;MAP KERNEL 1 TO THAT MEMORY
1589 005144 032701 000001          BIT    #1,R1        ;ERROR RETURN INDICATED?
1590 005150 001403                BEQ    .+10         ;NO- BRANCH
1591 005152 104002                ER^OR           ;MAP INDICATES NO MEMORY WAS FOUND
1592                                ;CORRESPONDING TO THIS REGISTER
1593                                ;R0 POINTS TO THE ADDRESS OF THE
1594                                ;PARITY REGISTER
1595 005154 000167 177744          JMP    LOOP13
1596 005160 012737 005214 000114  MOV    #TRP13,@#PARVEC ;SETUP PARITY TRAP RETURN
1597 005166 012770 000004 000000  MOV    #WWP,@(R0)     ;SET WRITE WRONG PARITY
1598 005174 012711 125252          MOV    #125252,@R1   ;WRITE WRONG PARITY
1599 005200 012770 000001 000000  MOV    #AE,@(R0)     ;SET ACTION ENABLE AND CLEAR
1600                                ;WRITE WRONG PARITY
1601 005206 005711                TST    @R1          ;DATI WITH ACTION ENABLE SET SHOULD
1602                                ;ABORT LEAVING DATA UNCHANGED
1603 005210 104002                ERROR           ;NO ABORT ON READING BAD PARITY
1604                                ;WITH ACTION ENABLE SET. R0 POINTS
1605                                ;TO THE ADDRESS OF THE PARITY REGISTER.
1606                                ;R1 CONTAINS THE ADDRESS OF THE TEST
1607                                ;LOCATION (VIRTUAL THRU KERNEL PAGE
1608                                ;1 IF KT11 IS PRESENT).
1609 005212 000434                TRP13: BR     CONT13
1610 005214 005070 000000          CLR    @(R0)          ;ABORT OCCURRED AS EXPECTED- CLEAR REGISTER
1611 005220 021127 125252          CMP    @R1,#125252   ;ORIGINAL DATA RESTORED?
1612 005224 001401                BEQ    .+4            ;YES, BRANCH
1613 005226 104002                ERROR           ;DATI WHICH GOT A PARITY ERROR ALTERED
1614                                ;THE CONTENTS OF THE LOCATION ADDRESSED.
1615                                ;(R1 CONTAINS THE ADDRESS OF
1616                                ;MEMORY BEING TESTED- IF KT11 IS
1617                                ;PRESENT, ADDRESS IN R1 IS VIRTUAL)
1618                                ;R0 POINTS TO THE ADDRESS OF THE
1619                                ;PARITY REGISTER
1620 005230 005770 000000          TST    @(R0)          ;CHECK PARITY REGISTER
1621 005234 100401                BMI    .+4            ;BRANCH IF PARITY ERROR SET
1622 005236 104002                ERROR           ;PARITY ERROR NOT SET AFTER READING
1623                                ;DATA WITH BAD PARITY
1624                                ;R0 POINTS TO THE ADDRESS OF THE PARITY
1625                                ;REGISTER. R1 CONTAINS THE ADDRESS
1626                                ;OF THE TEST LOCATION (VIRTUAL THRU
1627                                ;KERNEL PAGE 1 IF KT11 PRESENT)
1628 005240 022626                CMP    (SP)+,(SP)+   ;RESTORE STACK POINTER
1629 005242 012770 000004 000000  MOV    #WWP,@(R0)     ;SET WRITE WRONG PARITY, CLEAR PARITY ERROR
1630 005250 012711 125252          MOV    #125252,@R1   ;REWRITE DATA WITH WRONG PARITY
1631 005254 005070 000000          CLR    @(R0)          ;CLEAR PARITY REGISTER
1632 005260 012737 000116 000114  MOV    #PARVEC+2,@#PARVEC ;RESTORE TRAPCATCHER
1633 005266 005711                TST    @R1          ;DATI TO LOCATION WITH BAD PARITY
1634                                ;AE NOT SET-INSTRUCTION SHOULD COMPLETE
1635 005270 005070 000000          CLR    @(R0)          ;CLEAR PARITY ERROR BIT
1636 005274 022711 125252          CMP    #125252,@R1   ;CHECK DATA
1637 005300 001401                BEQ    .+4            ;
1638 005302 104002                ERROR           ;DATI TO LOCATION WITH BAD PARITY
1639                                ;WITHOUT ACTION ENABLE SET LEFT INCORRECT DATA
1640                                ;R1 CONTAINS THE ADDRESS OF THE TEST
1641                                ;LOCATION (VIRTUAL THRU KERNEL PAGE 1

```



```

1754 005770 103765          BLO      1$
1755 005772 104002          ERROR
1756 005774 000405          BR      3$
1757
1758 005776 011167 172764    2$:    MOV     @R1,TREG
1759 006002 104004          ERRORS
1760
1761
1762
1763
1764 006004 004767 006504          JSR     %7,PSCAN
1765
1766 006010 022626          3$:    CMP     (SP)+,(SP)+
1767 006012 000207          RTS     %7
1768
1769
1770
1771
1772
1773
1774
1775
1776 006014 104001          ;*****
1777 006016 012777 000015 173056    MOV     #15,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
1778 006024 005767 172520          TST     NDKT ;KT11 PRESENT?
1779 006030 001402          BEQ     .+6 ;YES- BRANCH
1780 006032 000167 000424          JMP     TEST16 ;NO, SKIP TEST
1781 006036 004767 005750          JSR     PC,CLRPAR ;CLEAR ALL PARITY REGISTERS
1782 006042 004767 005644          JSR     %7,NRALL ;MAP KERNEL 0 TO BANK 0,RW
1783 006046 004767 006010          JSR     %7,MAP1 ;MAP KERNEL 7 TO THE EXTERNAL BANK
1784
1785 006052 012777 001600 173172    MOV     #1600,@KPAR1 ;SET KERNEL 1 RW AND TURN ON KT11
1786 006060 005067 172470          CLR     LOWFLG ;MAP KERNEL PAGE 1 TO BEGINNING OF 28-32K
1787 006064 016767 173000 173002    MOV     PMEMH,PMEMX ;CLEAR FLAG TO INDICATE CHECKING LOWER 64K
1788 006072 012737 006402 000114    MOV     #TRP15,@#PARVEC ;SETUP PARITY TRAP RETURN
1789 006100 012767 000200 172412    MOV     #200,BITPT ;INITIALIZE BIT POINTER
1790 006106 036767 172406 172760    LOOP15: BIT BITPT,PMEMX ;DOES THIS 4K HAVE PARITY?
1791 006114 001022          BNE     TST15 ;YES, BRANCH TO TEST IT
1792 006116 062777 000200 173126    LOP15: ADD #200,@KPAR1 ;NO- MAP TO NEXT 4K
1793 006124 006367 172370          ASL     BITPT ;UPDATE BIT POINTER
1794 006130 103366          BCC     LOOP15 ;BRANCH IF NOT DONE WITH 64K
1795 006132 005767 172416          TST     LOWFLG ;DONE WITH 128K?
1796 006136 001051          BNE     DONE15 ;YES, BRANCH
1797 006140 005267 172410          INC     LOWFLG ;NO, SET FLAG INDICATING UPPER 64K
1798 006144 016767 172722 172722    MOV     PMEMH,PMEMX ;SETUP PARITY MAP WORD
1799 006152 012767 000001 172340    MOV     #1,BITPT ;SETUP BIT POINTER FOR UPPER 64K
1800 006160 000752          BR      LOOP15 ;CONTINUE
1801 006162 012704 001036          TST15: MOV #PARPAT,R4 ;INITIALIZE PATTERN POINTER
1802 006166 012767 020000 172322    MOV     #20000,ADRPT ;INITIALIZE VIRTUAL ADDRESS OF MEMORY
1803
1804 006174 012705 020000          MOV     #20000,R5 ;BEING TESTED
1805 006200 005025          CLR     (5)+ ;INITIALLY CLEAR CORE BLOCK UNDER TEST
1806 006202 020527 040000          CMP     R5,#40000
1807 006206 103774          BLO     2$
1808 006210 012701 000566          MOV     #MPRO,R1 ;INITIALIZE TO SET ACTION ENABLE IN ALL
1809
    
```

```

1810 006214 032711 000001          3$:    BIT     #1,@R1
1811 006220 001003          BNE     .+10
1812 006222 012771 000001 000000    MOV     #AE,@(R1) ;SET ACTION ENABLE IF THIS REGISTER
1813
1814 006230 062701 000010          ADD     #10,R1 ;IS PRESENT
1815 006234 020127 000766          CMP     R1,#TREG
1816 006240 103765          BLO     3$
1817 006242 004767 000030          4$:    JSR     %7,TPCORX ;EXERCISE THIS 4K
1818 006246 005724          TST     (4)+ ;UPDATE PATTERN
1819 006250 005714          TST     (4) ;LAST PATTERN?
1820 006252 001373          BNE     4$ ;NO, LOOP
1821 006254 004767 005532          JSR     PC,CLRPAR ;YES, CLEAR ALL PARITY REGISTERS
1822 006260 000716          BR      LOP15 ;UPDATE AND CHECK NEXT 4K
1823 006262 005037 177572          DONE15: CLR @#SR0 ;TURN OFF KT11 WHEN DONE
1824 006266 012737 000116 000114    MOV     #PARVEC+2,@#PARVEC ;STORE TRAPCATCHER
1825 006274 000472          BR      TEST16 ;GO TO NEXT TEST
1826
1827 ;PARITY MEMORY TEST ROUTINE USING KT11 AND TESTING MEMORY ABOVE 28K
1828 ;WRITES AND CHECKS EACH LOCATION IN 4K USING KERNEL PAGE 1 MAPPED TO CURRENT BANK
1829 006276 000240          TPCORX: NOP
1830 006300 012705 020000          MOV     #20000,R5 ;SETUP R5 TO POINT TO THE LOCATION
1831
1832 006304 011415          1$:    MOV     (4),(5) ;UNDER TEST (VIRTUAL ADDRESS)
1833 006306 011567 172224          MOV     (5),WAS ;WRITE PATTERN
1834 006312 021467 172220          CMP     (4),WAS ;READ TEST LOCATION
1835 006316 001401          BEQ     .+4 ;DATA OK?
1836 006320 104002          ERROR ;YES- BRANCH
1837
1838 ;NO- DATA INCORRECT IN LOCATION WHOSE
1839 ;VIRTUAL ADDRESS IS IN R1 (GOES THRU
1840 ;KERNEL PAGE 1). R4 POINTS TO
1841 ;THE VALUE WRITTEN.
1842 ;UPDATE ADDRESS POINTER
1843 006322 005725          TST     (5)+ ;THIS 4K DONE?
1844 006324 020527 040000          CMP     R5,#40000
1845 006330 103765          BLO     1$ ;NO, BRANCH TO TEST NEXT LOCATION
1846 006332 005067 172430          CLR     TREG ;YES, CHECK TO SEE IF ANY PARITY
1847
1848 ;ERRORS OCCURRED WITHOUT TRAPPING
1849 006336 012701 000566          MOV     #MPRO,R1
1850 006342 032711 000001          2$:    BIT     #1,(1) ;IS THIS PARITY REGISTER PRESENT?
1851 006346 001003          BNE     .+10 ;NO, GET NEXT ONE
1852 006350 005771 000000          TST     @(R1) ;YES- DID ERROR SET?
1853 006354 100406          BMI     3$ ;YES- BRANCH
1854 006356 062701 000010          ADD     #10,R1 ;NO- GET NEXT REGISTER
1855 006362 020127 000766          CMP     R1,#TREG
1856 006366 103765          BLO     2$
1857 006370 000207          RTS     %7 ;NO ERRORS- EXIT
1858 006372 011167 172370          3$:    MOV     @R1,TREG ;STORE ADDRESS OF REGISTER GETTING ERROR
1859 006376 104004          ERRORS ;PARITY ERROR SET (AE ALREADY SET)
1860
1861 ;AND NO TRAP OR TIMEOUT OCCURRED
1862 ;R1 POINTS TO THE ADDRESS OF THE
1863 ;PARITY REGISTER
1864 006400 000207          RTS     %7
1865
1866 ;PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
1867 TRP15: CLR TREG
1868 006402 005067 172360          MOV     #MPRO,R1 ;LOCATE PARITY REGISTER INDICATING ERROR
1869 006406 012701 000566
1870 006412 032711 000001          1$:    BIT     #1,(R1)
1871 006416 001003          BNE     .+10
    
```

```

1866 006420 005771 000000          TST      @(R1)
1867 006424 100407          BMI      2$
1868 006426 062701 000010          ADD      #10,R1          ;BRANCH IF PARITY ERROR IS SET
1869 006432 020127 000766          CMP      R1,#TREG
1870 006436 103765          BLO     1$
1871 006440 104002          ERROR
1872
1873 006442 000405          BR      3$
1874 006444 011167 172316          2$: MOV      @R1,TREG          ;STORE ADDRESS OF REGISTER GETTING ERROR
1875 006450 104004          ERRORS          ;PARITY TRAP TO 114 OCCURRED DUE TO
1876
1877
1878
1879 006452 004767 006036          JSR      %7,PSCAN          ;"TREG" CONTAINS ADDRESS OF PARITY REGISTER
1880
1881
1882
1883 006456 022626          3$: CMP      (SP)+,(SP)+          ;HAVING PARITY ERROR BIT SET
1884 006460 000207          RTS      %7          ;SCAN MEMORY FOR BAD PARITY AND PRINT 18
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896 006462 104001          TEST16: SCOPE
1897 006464 012777 000016 172410          MOV      #16,@DISPLAY          ;LOAD THE TEST NUMBER INTO THE DISPLAY
1898 006472 012737 007076 000114          MOV      #TRP16,@#PARVEC          ;SET UP TRAP RETURN
1899 006500 032767 000002 172012          BIT      #2,BITPT          ;TEST BANK INDICATOR FOR BANK 1
1900 006506 001007          BNE     LUP16
1901 006510 012767 000004 172002          MOV      #4,BITPT          ;INIT POINTER TO BANK 2
1902 006516 012767 040000 171772          MOV      #40000,ADRPT
1903 006524 000403          BR      -+10
1904 006526 012767 020000 171762          LUP16: MOV      #20000,ADRPT
1905 006534 036767 171760 172326          1$: BIT      BITPT,PMEML          ;DOES THIS 4K HAVE PARITY?
1906 006542 001021          BNE     3$
1907 006544 032767 000002 171746          2$: BIT      #2,BITPT          ;YES, BRANCH TO TEST IT
1908 006552 001001          BNE     -+4
1909 006554 000402          BR      -+6
1910 006556 000167 001546          JMP      OUT
1911 006562 062767 020000 171726          ADD      #20000,ADRPT          ;NO, UPDATE MEMORY ADDRESS BY 4K
1912 006570 006367 171724          ASL     BITPT
1913 006574 022767 000200 171716          CMP      #200,BITPT          ;UPDATE BIT POINTER
1914 006602 003354          BGT     1$
1915 006604 000421          BR      DONE16
1916 006606 016767 171704 171714          3$: MOV      ADRPT,HIADR          ;THIS 28K DONE?
1917 006614 062767 020000 171706          ADD      #20000,HIADR          ;NO, CHECK NEXT 4K
1918 006622 004767 005164          JSR      %7,CLRPAR          ;YES, EXIT
1919 006626 016705 171664          MOV      ADRPT,R5          ;SET UPPER LIMIT THIS 4K
1920 006632 005025          5$: CLR      (5)+
1921 006634 020567 171670          CMP      R5,HIADR          ;CLEAR ALL PARITY REGISTERS
          ;CLEAR BANK UNDER TEST
    
```

```

;*****
;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.
;*****
    
```

```

1922 006640 103774          BLO     5$
1923 006642 004767 000020          6$: JSR      %7,WWP16          ;GO WRITE WRONG PARITY IN EACH BYTE
1924 006646 000736          BR      2$          ;UPDATE AND CHECK NEXT 4K
1925 006650 004767 005136          DONE16: JSR      %7,CLRPAR          ;CLEAR ALL PARITY REGISTERS IF DONE
1926 006654 012737 000116          MOV      #PARVEC+2,@#PARVEC          ;RESTORE TRAP CATCHER
1927 006662 000167 000472          JMC     TEST17          ;GO TO NEXT TEST
1928
1929
1930
1931
1932 006666 016705 171624          ;WRITE WRONG PARITY TEST ROUTINE - TESTS EACH BYTE IN 4K
1933 006672 005067 171660          ;USING SAME DATA VALUE, WRITES AND CHECKS PARITY IN WRONG STATE
1934 006676 012767 125253 171630          ;AND THEN IN CORRECT STATE TO PROVE THAT PARITY BITS TOGGLE
1935 006704 012715 125253          WWP16: MOV      ADRPT,R5          ;SET TEST ADDRESS POINTER
1936 006710 012701 000566          CLR      ODDFLG          ;INDICATE TESTING LOW BYTE
1937 006714 032711 000001          MOV      #125253,SHDBE          ;STORE DATA FOR USE BY ERROR TYPFOUT ROUTINE
1938 006720 001003          WWP16A: MOV      #125253,@R5          ;INITIALIZE TEST LOCATION
1939 006722 012771 000005 000000          MOV      #MPRO,R1          ;SETUP TO LOAD PARITY REGISTERS
1940
1941
1942 006730 062701 000010          1$: BIT      #1,(1)
1943 006734 020127 000766          BNE     -+10
1944 006740 103765          MOV      #WWP+AE,@(R1)          ;SET WRITE WRONG PARITY AND ACTION
1945 006742 005767 171610          ;ENABLE IF THIS PARITY REGISTER
1946 006746 100425          ;IS PRESENT
1947 006750 112715 000253          ADD      #10,R1
1948 006754 012701 000566          CMP      R1,#TREG
1949 006760 032711 000001          BLO     1$
1950 006764 001003          TST     ODDFLG          ;WRITING HIGH BYTE?
1951 006766 042771 000004 000000          BMI     2$          ;YES, BRANCH
1952 006774 062701 000010          MOV      #253,@R5          ;NO, WRITE WRONG PARITY IN LOW BYTE
1953 007000 020127 000766          MOV      #MPRO,R1          ;THIS CODE CLEARS WWP BIT IN ALL
1954 007004 103765          BIT      #1,(1)          ;PARITY REGISTERS
1955 007006 005767 171544          BNE     -+10
1956 007012 100407          BIC     #WWP,@(R1)
1957 007014 142715 000377          ADD      #10,R1
1958
1959 007020 000407          ADD      #10,R1
1960 007022 112765 000252 000001          2$: MOV      #252,1(R5)
1961 007030 000751          BR      5$
1962 007032 142765 000377 000001          5$: BIC     #377,1(R5)          ;WRITE WRONG PARITY IN HIGH BYTE
1963
1964 007040 012701 000566          6$: BIC     #377,1(R5)          ;DETECT WRONG PARITY WITH DATIP-
1965
1966 007044 032711 000001          3$: MOV      #MPRO,R1          ;SHOULD TRAP TO TRP16 BEFORE DOING THE DATOB
1967 007050 001003          ;NO TRAP OCCURRED- SETUP TO CLEAR
1968 007052 042771 000005 000000          4$: BIT      #1,@R1          ;AE AND WWP
1969 007060 062701 000010          BNE     -+10
1970 007064 020127 000766          BIC     #WWP+AE,@(R1)          ;CLEAR AE AND WWP IN ALL PARITY REGISTERS
1971 007070 103765          ADD      #10,R1
1972 007072 104002          CMP      R1,#TREG
1973
1974
1975
1976
1977
          ERROR
          ;ERROR, NO TRAP AFTER WRITING AND
          ;READING WRONG PARITY IN LOCATION
          ;WHOSE ADDRESS IS IN R5 (AE AND WWP
          ;WERE SET IN ALL PARITY REGISTERS)
          ;TESTING LOW BYTE IF ODDFLG IS POSITIVE
          ;TESTING HIGH BYTE IF ODDFLG IS NEGATIVE
    
```

```

1978                                     ;NOTE THAT AE AND WWP WERE CLEARED
1979                                     ;BEFORE TYPING THE ERROR PRINTOUT
1980 007074 000517          BR          CN16
1981
1982                                     ;WHEN WRONG PARITY DATA IS READ BACK SHOULD ENTER HERE VIA TRAP TO 114
1983 007076 012701 000566  TRP16:  MOV    #MPRO,R1          ;PARITY TRAP OCCURRED- FIRST CLEAR
1984 007102 032711 000001  TRP16:  BIT    #1,@R1          ;WWW AND AE IN ALL REGISTERS
1985 007106 001003          BNE          .+10
1986 007110 042771 000005 000000  BIC    #AE+WWP,@(R1)
1987 007116 062701 000010  ADD    #10,R1
1988 007122 020127 000766  CMP    R1,#TREG
1989 007126 103765          BLO    TRP16A
1990 007130 012701 000566  MOV    #MPRO,R1          ;FIND THE REGISTER THAT SENSED THE ERROR
1991 007134 012703 000770  MOV    #INDCO,R3
1992 007140 005067 171622  CLR    TREG
1993 007144 032711 000001 1$:   BIT    #1,@R1          ;DOES THIS CONTROL EXIST?
1994 007150 001017          BNE          2$
1995 007152 005771 000000  TST   @(R1)          ;YES- IS ERROR SET?
1996 007156 100014          BPL          2$
1997 007160 005767 171602  TST   TREG          ;YES- WAS IT SET IN ANY OTHER REGISTER ALSO?
1998 007164 001401          BEQ          .+4
1999 007166 104002          ERROR
2000                                     ;ERROR SET IN MORE THAN ONE PARITY REGISTER
2001                                     ;AFTER WRITING WRONG PARITY IN LOCATION
2002 007170 036761 171324 000002  BIT    BITPT,2(R1)    ;WHOSE ADDRESS IS IN R5
2003                                     ;DOES MAP INDICATE THIS PARITY REGISTER
2004 007176 001001          BNE          .+4
2005 007200 104002          ERROR          ;CONTROLS THIS MEMORY?
2006                                     ;YES, BRANCH
2007                                     ;PARITY REGISTER RESPONDED TO MEMORY
2008                                     ;NOT INCLUDED IN ITS MAP
2009                                     ;PARITY REGISTER'S ADDRESS IS POINTED
2010 007202 010304          MOV    R3,R4          ;TO BY R1, ADDRESS OF LOCATION CAUSING
2011 007204 011167 171556  MOV    @R1,TREG        ;PARITY ERROR IS IN R5
2012 007210 062701 000010 2$:   ADD    #10,R1
2013 007214 005723          TST   (R3)+
2014 007216 020127 000766  CMP    R1,#TREG
2015 007222 103750          BLO    1$
2016                                     ;BRANCH UNTIL ALL THE PARITY
2017 007224 011567 171306  MOV    (5),WAS        ;REGISTERS HAVE BEEN CHECKED
2018 007230 022715 125253  CMP    #125253,@R5    ;SAVE DATA FROM LOCATION UNDER TEST
2019 007234 001401          BEQ          .+4
2020 007236 104003          ERRORP
2021                                     ;DID BICB CHANGE DATA?
2022                                     ;NO, CONTINUE
2023                                     ;DATA WAS MODIFIED BY THE BICB WHICH
2024                                     ;GOT A PARITY ERROR TRAP- SINCE PARITY ERROR
2025                                     ;TRAP OCCURRED, CONTENTS SHOULD NOT HAVE
2026                                     ;BEEN MODIFIED. R5 CONTAINS ADDRESS
2027 007240 005767 171522  TST   TREG          ;OF TEST LOCATION. "TREG" CONTAINS
2028 007244 001002          BNE          3$
2029 007246 104002          ERROR          ;ADDRESS OF PARITY REGISTER SENSING
2030                                     ;ERROR
2031                                     ;WAS PARITY ERROR SET IN ANY REGISTERS?
2032                                     ;YES- BRANCH
2033                                     ;PARITY TRAP OCCURRED ON READING
2034                                     ;WRONG PARITY (WITH AE SET) BUT NO
2035                                     ;REGISTERS HAD PARITY ERROR BIT SET.
2036                                     ;R5 CONTAINS THE ADDRESS OF THE
2037                                     ;TEST LOCATION.
    
```

```

2034 007250 000420          BR          4$
2035 007252 022714 000001 3$:   CMP    #1,(R4)
2036 007256 001015          BNE          4$
2037 007260 017701 171502  MOV    @TREG,R1          ;GET PARITY REGISTER CONTENTS
2038 007264 042701 170037  BIC    #170037,R1      ;MASK OFF ALL BUT ERROR ADDRESS BITS
2039 007270 010502  MOV    R5,R2          ;GET ADDRESS OF LOCATION UNDER TEST
2040 007272 042702 003777  BIC    #3777,R2        ;POSITION BITS IN R2
2041 007276 000302  SWAB   R2
2042 007300 006302  ASL   R2
2043 007302 006302  ASL   R2
2044 007304 020102  CMP    R1,R2          ;PARITY ERROR ADDRESS BITS CORRECT?
2045 007306 001401  BEQ    .+4
2046 007310 104003  ERRORP
2047                                     ;ERROR ADDRESS BITS (PARITY REGISTER
2048                                     ;BITS 11-5) ARE INCORRECT. R5 CONTAINS
2049                                     ;THE ADDRESS OF THE TEST LOCATION.
2050                                     ;"TREG" CONTAINS THE ADDRESS OF THE
2051 007312 022626 4$:   CMP    (SP)+,(SP)+    ;PARITY REGISTER DETECTING THE ERROR
2052 007314 011515  CNT16: MOV    (5),(5)    ;RESTORE STACK POINTER
2053 007316 005077 171444  CLR    @TREG          ;RESTORE TEST LOCATION TO FIX BAD PARITY
2054 007322 005715  TST   @R5            ;CLEAR ERROR BIT IN PARITY REGISTER
2055 007324 005777 171436  TST   @TREG          ;READ LOCATION TO SEE IF PARITY IS GOOD
2056 007330 100001  BPL   .+4            ;IS PARITY ERROR SET?
2057 007332 104002  ERROR          ;NO- BRANCH
2058                                     ;WRITING LOCATION WITH WRITE WRONG PARITY
2059                                     ;CLEAR DIDN'T CLEAR BAD PARITY
2060                                     ;R5 CONTAINS ADDRESS OF THE TEST
2061                                     ;LOCATION. "TREG" CONTAINS THE ADDRESS
2062                                     ;OF THE PARITY REGISTER DETECTING
2063 007334 005167 171216  CN16: COM   ODDFLG    ;TOGGLE BYTE INDICATOR
2064 007340 100401  BMI   .+4            ;BRANCH IF READY TO TEST HIGH BYTE
2065 007342 005725  TST   (5)+          ;UPDATE ADDRESS POINTER
2066 007344 020567 171160  CMP    R5,HIADR      ;THIS 4K DONE?
2067 007350 103401  BLO   1$            ;NO, TEST NEXT LOCATION
2068 007352 000207  RTS    %7            ;RETURN TO TEST NEXT BANK
2069 007354 000167 177324 1$:   JMP    WWP16A
2070
2071
2072
2073                                     ;*****
2074                                     ;FORCE WRONG PARITY IN EACH BYTE OF PARITY MEMORY ABOVE 28K
2075                                     ;WRITE WRONG PARITY AND READ IT WITH ACTION ENABLE SET, MAKING SURE
2076                                     ;THAT A TRAP OCCURS. THEN WRITE AND READ THE SAME LOCATION WITH GOOD PARITY
2077                                     ;(USING SAME DATA) TO SHOW THAT THE PARITY BIT TOGGLES. MAKE SURE THAT
2078                                     ;THE ERROR ADDRESS BITS (PARITY REGISTER BITS 5-11) ARE CORRECT.
2079                                     ;** *****
2080 007360 104001  TEST17: SCOPE
2081 007362 004767 005032  JSR   PC,RSTLDR
2082 007366 012777 000017 171506  MOV    #17,@DISPLAY
2083 007374 012777 000017 171500  MOV    #17,@DISPLAY    ;LOAD THE TEST NUMBER INTO THE DISPLAY
2084 007402 005767 171142  TST   NDKT          ;KT11 PRESENT?
2085 007406 001402  BEQ   .+6            ;YES, BRANCH
2086 007410 000167 000636  JMP   XFR1          ;NO, SKIP TO NEXT TEST
2087 007414 004767 004372  JSR   PC,CLRPAR     ;CLEAR ALL PARITY REGISTERS
2088 007420 012737 007762 000114  MOV    #TRP17,@#PARVEC ;SETUP FG. PARITY TRAP
2089 007426 012767 000200 171064  MOV    #200,BITPT    ;INITIALIZE 4K BIT POINTER
    
```

```

2090 007434 004767 004252 JSR %7,NRALL ;INITIALIZE ALL PAGES TO NON-RESIDENT
2091 007440 004767 004416 JSR %7,MAP1 ;MAP KERNEL 0 TO BANK 0,RW; KERNEL 7
2092 ;TO THE EXTERNAL BANK, RW. TURN ON
2093 ;THE KT11 AND MAKE KERNEL 1 RW
2094 007444 012777 001600 171600 MOV #1600,@KPAR1 ;INITIALIZE KERNEL PAGE 1 TO 2BK
2095 007452 005067 171076 CLF LOWFLG ;CLEAR FLAG TO INDICATE TESTING LOWER 64K
2096 007456 016767 171406 171410 MOV PMEMH,PMEMX
2097 007464 012767 000002 000366 MOV #2,INDX17
2098 007472 036767 171022 171374 1$: BIT BITPT,PMEMX ;SETUP OFFSET TO CHECK LOWER 64K OF MAPS
2099 007500 001025 BNE 3$ ;DOES THIS 4K HAVE PARITY?
2100 007502 062777 000200 171542 2$: ADD #200,@KPAR1 ;YES, BRANCH TO TEST IT
2101 007510 006367 171004 ASL BITPT ;NO, MAP TO NEXT 4K
2102 007514 103366 BCC 1$ ;UPDATE BIT POINTER
2103 007516 005767 171032 TST LOWFLG ;GO CHECK TO SEE IF THIS 4K HAS PARITY
2104 007522 001025 BNE DONE17 ;END OF 128K?
2105 007524 005267 171024 INC LOWFLG ;YES, EXIT
2106 007530 016767 171336 171336 MOV PMEMH,PMEMX ;NO, SET FLAG TO INDICATE SHIFT TO
2107 007536 012767 000001 170754 MOV #1,BITPT ;HIGH 64K AND CHANGE MAPS
2108 007544 012767 000004 000306 MOV #4,INDX17
2109 007552 000747 BR 1$ ;SETUP OFFSET TO CHECK UPPER 64K OF MAPS
2110 007554 012705 020000 3$: MOV #20000,R5
2111 007560 005025 5$: CLR (5)+ ;CLEAR BANK UNDER TEST
2112 007562 020527 040000 CMP R5,#40000
2113 007566 103774 BLO 5$
2114 007570 004767 000020 6$: JSR %7,WVP17 ;GO WRITE WRONG PARITY AND CHECK IT
2115 007574 000742 BR 2$ ;UPDATE AND CHECK NEXT 4K
2116 007576 005037 177572 DONE17: CLR @#SRO ;TURN OFF KT11 WHEN DONE
2117 007602 012737 000116 000114 MOV #PARVEC+2,@#PARVEC ;RESTORE TRAP CATCHER
2118 007610 000167 000436 JMP XFR1 ;GO TO SETUP FOR NEXT TEST
2119
2120 ;WRITE WRONG PARITY TEST ROUTINE TO TEST MEMORY ABOVE 28K
2121 007614 012705 020000 WVP17: MOV #20000,R5 ;SET TEST ADDRESS POINTER
2122 007620 005067 170732 CLR ODDFLG ;CLEAR FLAG TO INDICATE TESTING LOW BYTE
2123 007624 012767 125253 170702 MOV #125253,SHOBE ;STORE DATA FOR USE BY ERROR TYPEDOUT ROUTINE
2124 007632 012715 125253 WVP17A: MOV #125253,@R5 ;INITIALIZE LOCATION
2125 007636 012701 000566 MOV #MPRO,R1 ;INITIALIZE REGISTER ADDRESS POINTER
2126 007642 032711 000001 1$: BIT #1,(1) ;DOES THIS CONTROL EXIST?
2127 007646 001003 BNE .+10 ;NO, GET NEXT
2128 007650 012771 000005 000000 MOV #WVP+AE,@(R1) ;YES- SET WRITE WRONG PARITY
2129 ;AND ACTION ENABLE
2130 007656 062701 000010 ADD #10,R1
2131 007662 020127 000766 CMP R1,#TREG ;ALL REGISTERS SETUP?
2132 007666 103765 BLO 1$ ;NO- LOOP
2133 007670 005767 170662 TST ODDFLG ;YES- TESTING HIGH BYTE?
2134 007674 100405 BMI 2$ ;YES, BRANCH
2135 007676 112715 000253 MOVB #253,@R5 ;NO, WRITE WRONG PARITY IN LOW BYTE
2136 007702 142715 000377 BICB #377,@R5 ;DETECT WRONG PARITY WITH DATIP-
2137 ;SHOULD TRAP TO TRP17 BEFORE DOING THE DATAB
2138 007706 000406 BR 3$
2139 007710 112765 000252 000001 2$: MOVB #252,1(R5) ;WRITE WRONG PARITY IN HIGH BYTE
2140 007716 142765 000377 000001 BICB #377,1(R5) ;DETECT WRONG PARITY WITH DATIP
2141 ;SHOULD TRAP TO TRP17 BEFORE DOING THE DATAB
2142 007724 012701 000566 3$: MOV #MPRO,R1 ;IF NO TRAP, CLEAR AE AND WVP IN ALL
2143 007730 052711 000001 4$: BIT #1,@R1 ;PARITY REGISTERS
2144 007734 001003 BNE .+10
2145 007736 042771 000005 000000 BIC #AE+WVP,@(R1)
    
```

```

2146 007744 062701 000010 ADD #10,R1
2147 007750 020127 000766 CMP R1,#TREG
2148 007754 103765 BLO 4$
2149 007756 104002 ERROR ;NO TRAP AFTER WRITING AND READING
2150 ;WRONG PARITY WITH AE SET- VIRTUAL
2151 ;ADDRESS OF LOCATION IS IN R5
2152 ;(MAPPED THRU KERNEL PAGE 1)
2153 ;WROTE LOW BYTE IF ODDFLG IS POSITIVE
2154 ;WROTE HIGH BYTE IF IT IS NEGATIVE
2155 ;NOTE THAT WVP AND AE WERE CLEARED
2156 ;BEFORE ERROR PRINTOUT
2157 007760 000512 BR CNT17
2158
2159 ;WHEN WRONG PARITY DATA IS READ BACK, SHOULD ENTER HERE VIA TRAP TO 114
2160 007762 012701 000566 TRP17: MOV #MPRO,R1 ;PARITY TRAP OCCURRED- BEFORE CHECKING
2161 007766 032711 000001 TRP17A: BIT #1,@R1 ;IT, CLEAR WVP AND AE IN ALL REGISTERS
2162 007772 001003 BNE .+10
2163 007774 042771 000005 000000 BIC #AE+WVP,@(R1)
2164 010002 062701 000010 ADD #10,R1
2165 010006 020127 000766 CMP R1,#TREG
2166 010012 103765 BLO TRP17A ;FIND REGISTER THAT SENSED THE ERROR
2167 010014 012701 000566 MOV #MPRO,R1 ;SETUP PTR TO INDICATOR
2168 010020 012703 000770 MOV #INDCO,R3
2169 010024 005067 170736 CLR TREG
2170 010030 032711 000001 1$: BIT #1,@R1 ;DOES THIS CONTROL EXIST?
2171 010034 001017 BNE 2$ ;NO, BRANCH
2172 010036 005771 000000 TST @(R1) ;YES- IS ERROR SET?
2173 010042 100014 BPL 2$ ;NO, BRANCH
2174 010044 005767 170716 TST TREG ;YES- WAS IT SET IN ANY OTHER REGISTER ALSO?
2175 010050 001401 BEQ .+4 ;NO- BRANCH
2176 010052 104002 ERROR ;ERROR SET IN MORE THAN ONE PARITY REGISTER
2177 ;AFTER READING WRONG PARITY IN LOCATION
2178 ;WHOSE VIRTUAL ADDRESS IS IN R5
2179 010054 036761 170440 000002 BIT BITPT,2(R1) ;DOES MAP INDICATE THAT THIS REGISTER
2180 ;CONTROLS THIS MEMORY?
2181 INDX17=-.2
2182 010062 001001 BNE .+4 ;YES- BRANCH
2183 010064 104002 ERROR ;PARITY REGISTER RESPONDED TO MEMORY
2184 ;NOT INCLUDED IN ITS MAP. R1 POINTS TO
2185 ;THE PARITY REGISTER'S ADDRESS, R5
2186 ;CONTAINS VIRTUAL ADDRESS OF THE LOCATION
2187 ;BEING TESTED (MAPPED THRU KERNEL
2188 ;PAGE 1)
2189 010066 011167 170674 MOV @R1,TREG ;STORE REGISTER ADDRESS
2190 010072 010304 MOV R3,R4 ;STORE PTR TO INDICATOR
2191 010074 062701 000010 2$: ADD #10,R1
2192 010100 005723 TST (R3)+ ;INCREMENT PTR TO INDICATOR
2193 010102 020127 000766 CMP R1,#TREG
2194 010106 103750 BLO 1$ ;LOOP UNTIL ALL THE PARITY REGISTERS
2195 ;HAVE BEEN CHECKED
2196 010110 011567 170422 MOV (5),WAS ;SAVE CONTENTS OF LOCATION UNDER TEST
2197 010114 022715 125253 CMP #125253,@R5 ;DID BICB CHANGE DATA?
2198 010120 001401 BEQ .+4 ;NO, CONTINUE
2199 010122 104002 ERROR ;DATA WAS MODIFIED BY THE BICB WHICH
2200 ;GOT A PARITY ERROR TRAP- SINCE PARITY
2201 ;TRAP OCCURRED, CONTENTS SHOULD NOT
    
```

```

2202
2203
2204
2205
2206 010124 005767 170636 TST TREG
2207 010130 001003 BNE 3$
2208 010132 104002 ERROR
2209
2210
2211
2212
2213
2214 010134 000423 BR 4$
2215 010136 001022 BNE 4$
2216 010140 022714 000001 3$: CMP #1,(R4)
2217 010144 001017 BNE 4$
2218
2219
2220
2221
2222 010146 017701 170614 MOV @TREG,R1
2223 010152 042701 170037 BIC #170037,R1
2224 010156 010502 MOV R5,R2
2225 010160 042702 163777 BIC #163777,R2
2226 010164 000302 SWAB R2
2227 010166 006302 ASL R2
2228 010170 006302 ASL R2
2229 010172 067702 171054 ADD @KPAR1,R2
2230 010176 020102 CMP R1,R2
2231 010200 001401 BEQ .+4
2232 010202 104004 ERRORS
2233
2234
2235
2236
2237
2238 010204 022626 4$: CMP (SP)+,(SP)+
2239 010206 011515 CNT17: MOV (5),(5)
2240 010210 005077 170552 CLR @TREG
2241 010214 005715 TST @R5
2242 010216 005777 170544 TST @TREG
2243 010222 100001 BPL .+4
2244 010224 104002 ERROR
2245
2246
2247
2248
2249
2250 010226 005167 170324 COM ODDFLG
2251 010232 100401 BMI .+4
2252 010234 005725 TST (5)+
2253 010236 020527 040000 CMP R5,#40000
2254 010242 103401 BLO 1$
2255 010244 000207 RTS %7
2256
2257 010246 000167 177360 1$: JMP WWP17A

```

```

;HAVE BEEN MODIFIED. R5 CONTAINS TEST
;LOCATION ADDRESS (VIRTUAL, MAPPED THRU
;KERNEL PAGE 1). "TREG" CONTAINS ADDRESS OF
;PARITY REGISTER DETECTING PARITY ERROR
;WAS PARITY ERROR SET IN ANY REGISTER?
;YES- BRANCH
;PARITY TRAP OCCURRED ON READING
;WRONG PARITY WITH AE SET BUT NO
;REGISTER HAS THE PARITY ERROR BIT
;SET. R5 CONTAINS VIRTUAL ADDRESS
;OF THE TEST LOCATION (MAPPED THRU
;KERNEL PAGE 1)
;NO, BRANCH. DO NOT CHECK THE
;IS THIS A CORE PARITY REGISTER?
;NO, BRANCH. DO NOT CHECK THE
;ADDRESS BITS OF THE LAST PARITY
;ERROR
;IF CORE REG, CHECK IF ADDRESS
;BITS OF LAST PAR ERR WERE STORED
;GET CONTENTS OF REGISTER DETECTING PARITY ERROR
;MASK OFF ALL BUT ADDRESS BITS
;CALCULATE TOP 7 BITS OF ERROR ADDRESS
;PARITY ERROR ADDRESS BITS CORRECT?
;PARITY ERROR ADDRESS BITS (PARITY
;REGISTER BITS 11-5) INCORRECT
;"TREG" CONTAINS ADDRESS OF PARITY
;REGISTER. R5 CONTAINS THE VIRTUAL
;ADDRESS OF THE TEST LOCATION
;(MAPPED THRU KERNEL PAGE 1)
;RESTORE STACK POINTER
;RESTORE TEST LOCATION TO FIX BAD PARITY
;CLEAR ERROR BIT IN THE PARITY REGISTER
;READ LOCATION TO SEE IF PARITY IS GOOD
;IS PARITY ERROR SET?
;NO, BRANCH
;WRITING LOCATION WITH WRITE WRONG
;PARITY CLEAR DIDN'T CLEAR BAD PARITY
;"TREG" CONTAINS THE ADDRESS OF THE
;PARITY REGISTER. R5 CONTAINS THE
;VIRTUAL ADDRESS OF THE TEST LOCATION
;(MAPPED THRU KERNEL PAGE 1)
;TOGGLE BYTE INDICATOR
;BRANCH IF HIGH BYTE NOT YET TESTED
;UPDATE ADDRESS POINTER
;THIS 4K DONE?
;NO, TEST NEXT LOCATION
;YES, RETURN TO CHECK FOR NEXT
;BANK TO _E TESTED
;GO AND TEST NEXT LOCATION

```

```

2258 010252 104001 XFR1: SCOPE
2259 ;IF THE FIRST BANK (BANK 0) IS PARITY MEMORY, SETUP TO TEST IT
2260 ;COPY THE FIRST 4K TO THE SECOND 4K, MOVE THE STACK POINTER TO BANK 1,
2261 ;AND THEN JUMP TO THE COPY OF TEST20 IN BANK 1
2262 010254 012767 000002 170236 MOV #2,BITPT ;XXXX
2263 010262 000167 175052 JM" TEST14 ;XXX
2264 010266 005737 000042 TST @#42
2265 010272 001402 BEQ .+6
2266 010274 000167 001324 JMP DONE ;IF THE PROGRAM IS RUNNING UNDER ACT THEN
2267 ;GO TO DONE
2268 010300 032767 000001 170562 BIT #1,PMEML
2269 010306 001002 BNE .+6 ;BRANCH IF YES
2270 010310 000167 001310 JMP DONE ;NO- DONE WITH TEST
2271 010314 032767 000002 170542 BIT #2,MEML ;IS THERE A SECOND 4K(BANK 1)?
2272 010322 001002 BNE .+6 ;YES, BRANCH
2273 010324 000167 001274 JMP DONE ;NO, EXIT
2274 010330 005037 177776 OUT: CLR @#PS ;CLEAR STATUS REGISTER
2275 010334 004767 003452 JSR %7,CLRPAR ;CLEAR ALL PARITY REGISTERS
2276 010340 012700 017770 MOV #17770,R0 ;R0 IS COUNTER TO MOVE 4K
2277 010344 005001 CLR R1 ;R1 POINTS TO LOCATION IN BANK 0
2278 010346 011161 020000 1$: MOV @R1,20000(R1) ;COPY FROM BANK 0 TO BANK 1
2279 010352 005721 TST (R1)+ ;MOVE POINTER
2280 010354 005300 DEC R0 ;DONE WITH 4K?
2281 010356 001373 BNE 1$ ;NO- BRANCH
2282 010360 062706 020000 ADD #20000,SP ;YES, MOVE STACK POINTER TO POINT TO BANK 1
2283 010364 012767 030462 026070 MOV #TEST20+20010,RETURN+20000 ;UPDATE SCOPE RETURN IN BANK 1
2284 010372 062767 020000 023232 ADD #20000,ERRA1+20000 ;UPDATE ADDRESSES USED IN ERROR TYPEOUT
2285 010400 062767 020000 023226 ADD #20000,ERRA2+20000
2286 010406 062767 020000 023230 ADD #20000,ERRA3+20000
2287 010414 062767 020000 023224 ADD #20000,ERRA4+20000
2288 010422 062767 020000 023226 ADD #20000,ERRA5+20000
2289 010430 022767 177570 170442 CMP #177570,SWR ;DOES THE CONSOLE HAVE A SWITCH REG. ?
2290 010436 001403 BEQ 2$ ;IF SO THEN GO TO 2$
2291 010440 062767 020000 010432 ADD #20000,SWR+20000 ;OTHERWISE THERE IS A SOFTWARE SWITCH LOCATION
2292 ;AND ITS ADDRESS SHOULD BE UPDATED
2293 010446 000167 020010 2$: JMP TEST20+20010 ;GO TO TEST 20 IN BANK 1
2294
2295
2296
2297
2298 ;*****
2299 ;IF FIRST 4K IS PARITY MEMORY, CHECK IT WITH A SERIES OF PATTERNS
2300 ;THIS SUBTEST IS RUN IN BANK 1 (20000 ABOVE THE ADDRESSES IN THE LISTING)
2301 ;*****
2301 010452 013746 177776 TEST20: MOV @#PS,-(SP) ;THESE 2 LINES DO THE SAME AS A SCOPE WITHOUT
2302 010456 004767 005674 JSR PC,SCOPEC ;USING AN EMT
2303 010462 012777 000020 170412 MOV #20,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
2304 010470 004767 003316 JSR %7,CLRPAR ;CLEAR ALL PARITY REGISTERS
2305 010474 012704 021036 MOV #PARPAT+20000,R4 ;INITIALIZE PATTERN POINTER
2306 010500 005005 CLR R5
2307 010502 005025 1$: CLR (5)+ ;INITIALLY CLEAR BANK 0
2308 010504 020527 020000 CMP R5,#20000
2309 010510 103774 BLO 1$
2310 010512 012737 030720 000114 MOV #TRP20+20000,@#PARVEC ;SETUP TRAP RETURN
2311 010520 012701 020566 MOV #MPRO+20000,R1 ;SETUP TO SET ACTION ENABLE IN ALL
2312 ;PARITY REGISTERS PRESENT
2313 010524 032711 000001 2$: BIT #1,@R1

```



```

2314 010530 001003          BNE      .+10
2315 010532 012771 000001 000000    MOV      #AE,@(R1)      ;SET ACTION ENABLE IF REGISTER IS PRESENT
2316 010540 062701 000010          ADD      #10,R1
2317 010544 020127 020766          CMP      R1,#TREG+20000
2318 010550 103765          BLO      2$              ;BRANCH UNTIL ALL REGISTER ADDRESSES
2319                                ;HAVE BEEN CHECKED
2320 010552 004767 000022          3$:     JSR      %7,CKBK0 ;EXERCISE THIS 4K
2321 010556 005724          TST      (4)+           ;UPDATE PATTERN
2322 010560 005714          TST      (4)            ;LAST PATTERN?
2323 010562 001373          BNE      3$             ;NO. LOOP
2324 010564 004767 003222          DONE20: JSR      PC,CLRPAR ;CLEAR ALL PARITY REGISTERS
2325 010570 012737 000116 000114    MOV      #PARVEC+2,@#PARVEC ;RESTORE TRAP CATCHER
2326 010576 000526          BR       TEST21         ;GO TO NEXT TEST
2327
    
```

```

2328                                ;PARITY MEMORY TEST ROUTINE
2329                                ;WRITES AND CHECKS EACH LOCATION IN BANK 0 (EXCEPT 114 AND 116)
2330                                ;WITH VALUE POINTED TO BY R4
2331 010600 005005          CKBK0:  CLR      R5              ;SET ADDRESS POINTER
2332 010602 011415          1$:     MOV      (4),(5)         ;WRITE PATTERN
2333 010604 021415          CMP      (4),(5)         ;DATA OK?
2334 010606 001404          BEQ      .+12            ;YES- BRANCH
2335 010610 013746 177776          MOV      @#PS,-(SP)      ;SETUP TO DO ERROR CALL VIA JSR
2336 010614 004767 002742          JSR      PC,ERR          ;ERROR- DATA INCORRECT IN LOCATION
2337                                ;WHOSE ADDRESS IS IN R5. R4 POINTS
2338                                ;TO THE VALUE WRITTEN.
2339                                ;UPDATE ADDRESS POINTER
2340 010620 005725          TST      (5)+           ;DON'T CHANGE CONTENTS OF 114 AND 116
2341 010622 020527 000114          CMP      R5,#114
2342 010626 001002          BNE      .+6
2343 010630 062705 000004          ADD      #4,R5
2344 010634 020527 020000          CMP      R5,#20000      ;HAS THE WHOLE BANK BEEN TESTED WITH
2345                                ;THIS PATTERN?
2346 010640 103760          BLO      1$             ;NO. BRANCH TO TEST NEXT LOCATION
2347 010642 005067 170120          CLR      TREG           ;YES, DID ANY PARITY ERROR BITS SET?
2348 010646 012701 020566          MOV      #MPRO+20000,R1
2349 010652 032711 000001          2$:     BIT      #1,(1)
2350 010656 001003          BNE      .+10
2351 010660 005771 000000          TST      @(R1)
2352 010664 100406          BMI      3$             ;YES- BRANCH
2353 010666 062701 000010          ADD      #10,R1
2354 010672 020127 020766          CMP      R1,#TREG+20000
2355 010676 103765          BLO      2$
2356 010700 000207          RTS
2357 010702 013746 177776          3$:     MOV      @#PS,-(SP)      ;NO- RETURN
2358 010706 004767 002650          JSR      PC,ERR          ;SETUP TO DO ERROR CALL VIA JSR
2359                                ;ERROR- PARITY ERROR BIT SET AND NO
2360                                ;PARITY TRAP OCCURRED
2361                                ;(AE WAS SET) - R1 POINTS TO ADDRESS
2362                                ;OF PARITY REGISTER
2363                                ;RESTORE STACK POINTER
2364                                ;
2365                                ;PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
2366 010720 005067 170042          TRP20:  CLR      TREG
2367 010724 012701 020566          MOV      #MPRO+20000,R1 ;FIND THE REGISTER RECORDING A PARITY ERROR
2368 010730 032711 000001          1$:     BIT      #1,(R1)
2369 010734 001003          BNE      .+10
2370 010736 005771 000000          TST      @(R1)
2371 010742 100412          BMI      2$             ;BRANCH IF ERROR IS SET
2372 010744 062701 000010          ADD      #10,R1
2373 010750 020127 020766          CMP      R1,#TREG+20000
2374 010754 103765          BLO      1$
2375 010756 013746 177776          MOV      @#PS,-(SP)      ;SETUP TO DO ERROR CALL VIA A JSR
2376 010762 004767 002574          JSR      PC,ERR          ;ERROR- PARITY TRAP TO 114 OCCURRED DURING
2377                                ;TEST 20 BUT NO REGISTERS HAVE PARITY
2378                                ;ERROR SET
2379 010766 000430          BR       5$
2380 010770 017102 000000          2$:     MOV      @(R1),R2 ;SAVE CONTENTS OF PARITY REGISTER
2381 010774 005003          CLR      R3              ;INITIALIZE ADDRESS POINTER
2382 010776 005071 000000          3$:     CLR      @(R1)      ;CLEAR PARITY REGISTER
2383 011002 005713          TST      @R3             ;READ LOCATION
2384 011004 005771 000000          TST      @(R1)          ;PARITY E-ROR SET?
2385 011010 100413          BMI      4$             ;YES- BRANCH
    
```

```

2384 011012 005723          TST      (R3)+          ;MOVE POINTER
2385 011014 020327 020000    CMP      R3,#20000
2386 011020 103766          BLO
2387 011022 010271 000000    MOV      R2,@(R1)      ;RESTORE PARITY REGISTER CONTENTS
2388 011026 013746 177776    MOV      @#PS,-(SP)    ;SETUP TO CALL ERROR VIA JSR
2389 011032 004767 002524    JSR     PC,ERR        ;PARITY ERROR OCCURRED WHILE TESTING
2390                                ;BANK 0 BUT NO BAD PARITY WAS FOUND
2391                                ;DURING SCAN OF BANK 0. R1 POINTS
2392                                ;TO THE ADDRESS OF THE PARITY REGISTER
2393                                ;DETECTING THE ERROR
2394 011036 000404          BR       5$
2395 011040 013746 177776    4$:     MOV      @#PS,-(SP)    ;SETUP TO DO ERROR CALL VIA JSR
2396 011044 004767 002512    JSR     PC,ERR        ;PARITY ERROR WHILE EXERCISING MEMORY
2397                                ;BANK 0- R1 POINTS TO ADDRESS OF PARITY
2398                                ;REGISTER DETECTING THE ERROR. R3 CONTAINS
2399                                ;THE ADDRESS OF THE LOCATION HAVING
2400                                ;BAD PARITY
2401 011050 022626          5$:     CMP      (SP)+,(SP)+
2402 011052 000207          RTS     %7            ;RETURN TO CHECK USING THE NEXT PATTERN
2403
2404
2405
2406
2407                                ;*****
2408                                ;FORCE WRONG PARITY IN EACH LOCATION IN BANK 0
2409                                ;NOTE THAT THIS SUBTEST IS EXECUTED IN BANK 1 (20000 ABOVE ADDRESSES
2410                                ;IN THE LISTING). MAKE SURE THAT WRONG PARITY IN EACH BYTE CAN BE DETECTED,
2411                                ;AND THAT WHEN GOOD PARITY IS WRITTEN AND READ NO PARITY ERROR IS DETECTED.
2412                                ;CHECK ERROR ADDRESS BITS (PARITY REGISTER BITS 5-11)
2413                                ;*****
2413 011054 013746 177776    TEST21: MOV     @#PS,-(SP)    ;SAME AS SCOPE WITHOUT DOING EMT
2414 011060 004767 005272    JSR     PC,SCOPE
2415 011064 012777 000021    MOV      #21,@DISPLAY    ;LOAD THE TEST NUMBER INTO THE DISPLAY
2416 011072 004767 002714    JSR     PC,CLRPAR        ;CLEAR ALL PARITY REGISTERS
2417 011076 005005          CLR      R5
2418 011100 005025          15:     CLR      (R5)+
2419 011102 020527 020000    CMP      R5,#20000
2420 011106 103774          BLO     1$
2421
2422                                ;WRITE WRONG PARITY TEST ROUTINE
2423                                ;USING SAME DATA VALUE, WRITES AND CHECKS PARITY IN WRONG STATE
2424                                ;AND THEN IN CORRECT STATE TO PROVE THAT PARITY BITS TOGGLE
2425 011110 005005          WWP21:  CLR      R5
2426 011112 005067 167440    CLR      ODDFLG          ;CLEAR FLAG TO INDICATE TESTING LOW BYTE
2427 011116 012767 125253    MOV      #125253,SHDBE   ;STORE DATA FOR USE BY ERROR TYPEOUT ROUTINE
2428 011124 012715 125253    WWP21A: MOV     #125253,@R5  ;INITIALIZE LOCATION
2429 011130 012701 020566    MOV      #MPRO+20000,R1  ;SETUP TO SET WRITE WRONG PARITY
2430                                ;IN ALL REGISTERS
2431 011134 032711 000001    1$:     BIT      #1,(1)
2432 011140 001003          BNE     .+10
2433 011142 012771 000004    MOV      #WWP,@(R1)      ;SET WRITE WRONG PARITY
2434 011150 062701 000010    ADD     #10,R1
2435 011154 020127 020766    CMP     R1,#TREG+20000
2436 011160 103765          BLO     1$
2437 011162 005767 167370    TST     ODDFLG          ;TESTING HIGH BYTE?
2438 011166 100404          BMI     2$
2439 011170 112715 000253    MOV     #253,@R5        ;YES, BRANCH
                                ;NO, WRITE WRONG PARITY IN LOW BYTE
    
```

```

2440 011174 005715          TST     (R5)            ;DETECT WRONG PARITY WITH DAT1
2441 011176 000405          BR      3$
2442 011200 112765 000252 000001 2$:     MOV     #252,1(R5)      ;WRITE WRONG PARITY IN HIGH BYTE
2443 011206 105765 000001    TST     1(R5)            ;DETECT WRONG PARITY WITH DAT1
2444 011212 012701 020566    3$:     MOV     #MPRO+20000,R1 ;SETUP TO CLEAR WWP IN ALL PARITY REGISTERS
2445 011216 012703 000770    MOV     #INDCO,R3        ;POINTER TO INDICATOR
2446 011222 032711 000001    4$:     BIT     #1,@R1
2447 011226 001003          BNE     .+10
2448 011230 042771 000004 000000    BIC     #WWP,@(R1)      ;CLEAR WWP IN ALL PARITY REGISTERS
2449 011236 062701 000010    ADD     #10,R1
2450 011242 020127 020766    CMP     R1,#TREG+20000
2451 011246 103765          BLO     4$
2452 011250 012701 020566    MOV     #MPRO+20000,R1  ;CHECK TO SEE IF ANY REGISTER DETECTED
2453 011254 005067 167506    CLR     TREG            ;THE PARITY ERROR
2454 011260 032711 000001    LOOP21: BIT    #1,@R1
2455 011264 001024          BNE     5$
2456 011266 005771 000000    TST     @(R1)
2457 011272 100021          BPL     5$
2458 011274 005767 167466    TST     TREG
2459 011300 001404          BEQ     .+12
2460 011302 013746 177776    MOV     @#PS,-(SP)      ;SETUP TO DO ERROR CALL VIA A JSR
2461 011306 004767 002250    JSR     PC,ERR        ;ERROR- PARITY ERROR SET IN MORE THAN
                                ;ONE REGISTER AFTER WRITING WRONG PARITY
2462                                ;IN LOCATION WHOSE ADDRESS IS IN R5
2463                                ;DOES MAP INDICATE THAT THIS REGISTER
2464 011312 032761 000001 000002    BIT     #1,2(R1)
2465                                ;CONTROLS THIS MEMORY?
2466                                ;YES, BRANCH
2467 011320 001004          BNE     .+12
2468 011322 013746 177776    MOV     @#PS,-(SP)      ;NO- SETUP TO DO ERROR CALL VIA A JSR
2469 011326 004767 002230    JSR     PC,ERR        ;ERROR- PARITY REGISTER RESPONDED
2470                                ;TO MEMORY WHICH IS NOT IN ITS MAP.
2471                                ;R1 POINTS TO PARITY REGISTER'S ADDRESS.
2472                                ;R5 CONTAINS ADDRESS OF LOCATION BEING
2473                                ;TESTED
2474                                ;STORE REGISTER ADDRESS
2473 011332 011167 167430    5$:     MOV     @R1,TREG
2474 011336 062701 000010    ADD     #10,R1
2475 011342 005723          TST     (R3)+
2476 011344 020127 020766    CMP     R1,#TREG+20000
2477 011350 103743          BLO     LOOP21
2478                                ;BRANCH UNTIL ALL REGISTERS HAVE BEEN
2479                                ;CHECKED
2480 011352 005767 167410    TST     TREG
2481 011356 001005          BNE     6$
2482 011360 013746 177776    MOV     @#PS,-(SP)      ;NO- SETUP TO DO ERROR CALL VIA A JSR
2483 011364 004767 002172    JSR     PC,ERR        ;ERROR- NO REGISTER HAS PARITY ERROR
2484                                ;SET AFTER READING WRONG PARITY IN LOCATION
2485                                ;WHOSE ADDRESS IS IN R5
2485 011370 000423          BR      7$
2486 011372 022713 000001    6$:     CMP     #1,(R3)
2487 011376 001020          BNE     7$
2488                                ;IS THIS A CORE PAR REG?
2489                                ;NO, BRANCH(MOS PAR REG DOES NOT
2490                                ;STORE ADDR BITS OF PAR ERROR)
2491 011400 017701 167362    MOV     @TREG,R1
2492 011404 042701 170037    BIC     #170037,R1
2493 011410 010502          MOV     R5,R2
2494 011412 042702 003777    BIC     #3777,R2
2495 011416 000302          SWAB   R2
2496 011420 006302          ASL    R2
2497 011422 006302          ASL    R2
    
```

```

2496 011424 020102      CMP      R1,R2      ;PARITY ERROR ADDRESS BITS CORRECT?
2497 011426 001404      BEQ      7$        ;BRANCH IF YES
2498 011430 013746      MOV      @#PS,-(SP) ;NO- SETUP TO DO ERROR CALL VIA A JSR
2499 011434 004767 002122 JSR      PC,ERR    ;ERROR- ADDRESS BITS (PARITY REGISTER
2500                                ;BITS 5-11) INCORRECT- ADDRESS OF PARITY
2501                                ;REGISTER IS CONTAINED IN LOCATION "TREG"
2502                                ;ADDRESS OF TEST LOCATION IS IN R5
2503 011440 011515      7$:  MOV      @R5,@R5  ;RESTORE TEST LOCATION TO FIX BAD PARITY
2504 011442 005077 167320 CLR      @TREG     ;CLEAR ERROR BIT IN PARITY REGISTER
2505 011446 005715      TST      @R5      ;READ LOCATION TO SET IF PARITY IS GOOD
2506 011450 005777 167312 TST      @TREG     ;CHECK PARITY ERROR BIT
2507 011454 100004      BPL      .+12     ;BRANCH IF NOT SET
2508 011456 013746 177776 MOV      @#PS,-(SP) ;SETUP TO DO ERROR CALL VIA A JSR
2509 011462 004767 002074 JSR      PC,ERR    ;ERROR- WRITING LOCATION WITH WRITE
2510                                ;WRONG PARITY CLEAR DIDN'T CLEAR BAD
2511                                ;PARITY (ADDRESS OF LOCATION IS IN R5)
2512                                ;"TREG" +20000 CONTAINS THE ADDRESS
2513                                ;OF THE PARITY REGISTER
2514 011466 005167 167064      COM      ODDFLG   ;TOGGLE BYTE INDICATOR
2515 011472 100401      BMI      .+4     ;BRANCH IF HIGH BYTE NOT YET TESTED
2516 011474 005725      TST      (R5)+   ;UPDATE ADDRESS POINTER
2517 011476 020527 020000      CMP      R5,#20000 ;THIS 4K DONE?
2518 011502 103610      BLO      WWP21A  ;LOOP TILL ALL 4K HAS BEEN TESTED
2519 011504 004767 002302 JSR      PC,CLRPAR ;CLEAR ALL PARITY REGISTERS
2520 011510 013746 177776 MOV      @#PS,-(SP) ;SETUP TO CALL SCOPE VIA JSR
2521 011514 004767 004636 JSR      PC,SCOPEC ;SCOPE
2522
2523                                ;COPY SECOND 4K BANK BACK TO FIRST 4K AND RETURN TO FIRST 4K BANK
2524 011520 012700 017770 XFR2:  MOV      #17770,R0 ;R0 IS USED AS A COUNTER
2525 011524 005001      CLR      R1      ;R1 POINTS TO THE CURRENT LOCATION
2526 011526 016111 020000      1$:  MOV      20000(R1),@R1 ;COPY BANK 1 TO BANK 0
2527 011532 005721      TST      (R1)+
2528 011534 005300      DEC      R0
2529 011536 001373      BNE      1$
2530 011544 162706 020000      SUB      #20000,SP ;RESTORE STACK POINTER
2531 011544 162737 020000 013632      SUB      #20000,@#ERRA1
2532 011552 162737 020000 013634      SUB      #20000,@#ERRA2
2533 011560 162737 020000 013644      SUB      #20000,@#ERRA3
2534 011566 162737 020000 013646      SUB      #20000,@#ERRA4
2535 011574 162737 020000 013656      SUB      #20000,@#ERRA5
2536 011602 022737 177570 001100      CMP      #177570,@#SWR ;DOES THE CONSOLE HAVE A SWITCH REG. ?
2537 011610 001403      BEQ      2$      ;IF SO THEN GO TO 2$
2538 011612 162737 020000 001100      SUB      #20000,@#SWR ;OTHERWISE RESTORE THE ADDRESS OF SOFTWARE
2539                                ;SWITCH REGISTER
2540 011620 000167 160000      2$:  JMP      DONE-20000 ;RETURN TO BANK 0
2541
2542
2543
2544
2545                                ;*****
2546                                ;AT THIS POINT EXECUTION RETURNS TO BANK 0
2547                                ;*****
2547 011624 012737 000116 000114 DONE:  MOV      #PARVEC+2,@#PARVEC ;RESTORE TRAPCATCHER
2548 011632 012706 000510      MOV      #STKPT,SP ;REINITIALIZE STACK POINTER
2549 011636 004767 002150      JSR      %7,CLRPAR ;CLEAR ALL PARITY REGISTERS
2550 011642 005237 011752      INC      @#PASANT ;KEEP TRACK OF PASSES COMPLETED
2551 011646 004567 003544      JSR      R5,OACNV
    
```

```

2552 011652 011752      PASCNT
2553 011654 017401      MPCNT
2554 011656 000006      6
2555 011660 104000      TYPE          ;TYPE BELL, "END PASS=" AND PASS COUNT
2556 011662 017363      MPGEN
2557 011664 005000      CLP          RO
2558 011666 005200      1$:  INC      RO          ;ALLOW TIME FOR END PASS MESSAGE TO PRINT
2559 011670 001376      BNE      1$
2560 011672 013705 000042      LOGICAL:  MOV      @#42,R5 ;LOADED BY MONITOR?
2561 011676 001405      BEQ      CONT      ;BRANCH IF NO
2562 011700 000005      RESET
2563 011702 004715      SENDAD: JSR      7,(5) ;GO TO MONITOR
2564 011704 000240      NOP
2565 011706 000240      NOP
2566 011710 000240      NOP
2567 011712 032777 000400 167160 CONT:  BIT      #BIT8,@SWR ;SWITCH 8 SET?
2568 011720 001401      BEQ      .+4
2569 011722 000000      HALT
2570 011724 105767 167162      TSTB     $TPFLG
2571 011730 001006      BNE      1$      ;IF NO TERMINAL, SKIP
2572 011732 105777 167146      TSTB     @TPS    ;WAIT FOR TTY TO FINISH SO THAT RESET
2573 011736 100375      BPL      .-4     ;WON'T CLOBBER THE BELL
2574 011740 112777 000000 167140      MDVB     #0,@TPB ;OUTPUT A NULL
2575 011746 000167 170134      1$:  JMP      BEGIN
2576 011752 000000      PASCNT:  0 ;PASS COUNT
2577 011754 000000      MONCNT:  0
2578
2579
2580
2581
2582                                ;*****
2583                                ;CREATE MAP INDICATING WHERE 4K BLOCKS OF MEMORY ARE PRESENT
2584                                ;*****
2585 011756 012737 012162 000004 MAPMEM: MOV      #MAPMB,@#4 ;SET NO MEM MANAGEMENT TRAP
2586 011764 005737 177572      TST      @#SRO   ;IS KT PRESENT? (TIMEOUT IF NO)
2587
2588                                ;MAP MEMORY USING KT11 - MAX OF 124K POSSIBLE
2589 011770 005067 166554      MAPMA:  CLR      NOKT ;INDICATE KT11 PRESENT
2590 011774 004767 001712      JSR      %7,NRALL ;INITIALLY SET ALL PAGES NONRESIDENT, BANK 0
2591 012000 004767 002056      JSR      %7,MAP1 ;MAP KERNEL 0 TO BANK 0, RW
2592                                ;MAP KERNEL 7 TO THE EXTERNAL BANK, RW
2593                                ;MAP KERNEL 1 RW, AND TURN ON KT11
2594 012004 005067 166534      CLR      TBANK
2595 012010 012767 177777 167046      MOV      #177777,MEML ;SET UP CORE MAPS
2596 012016 012767 077777 167042      MOV      #77777,MEMH
2597 012024 012767 000001 166466      MOV      #1,BITPT ;SET UP 4K POINTER
2598 012032 012767 001064 166506      MOV      #MEML,MEMUT
2599 012040 012737 012150 000004      MOV      #5$,@#4
2600 012046 016777 166472 167176 2$:  MOV      TBANK,@KPAR1 ;SET UP FOR TIME OUTS
2601 012054 005737 021000      TST      @#21000 ;MAP KERNEL PAGE 1 TO BANK BEING TESTED
2602 012060 005737 025000      TST      @#25000 ;1ST K PRESENT?
2603 012064 005737 031000      TST      @#31000 ;2ND K PRESENT?
2604 012070 005737 035000      TST      @#35000 ;3RD K PRESENT?
2605 012074 062767 000200 166442 3$:  ADD      #200,TBANK ;4TH K PRESENT?
2606 012102 006367 166412      ASL      BITPT  ;UPDATE TEST ADDRESS
2607 012106 103006      BCC      4$     ;UPDATE BANK POINTER
2608                                ;BRANCH IF NOT DONE WITH 64K SECTION
    
```

```

2608 012110 012767 000001 166402      MOV      #1,BITPT      ;YES, DO MEMH(64-124K)
2609 012116 012767 001066 166422      MOV      #MEMH,MEMUT
2610 012124 022767 007600 166412 4$:  CMP      #7600,TBANK
2611 012132 003345          BGT      2$
2612 012134 005037 177572      CLR      @#SRO
2613 012140 012737 000006 000004      MOV      #6,@#4
2614 012146 000207          RTS
2615 012150 046777 166344 166370 5$:  BIC      BITPT,@MEMUT
2616          ;TIMEOUT OCCURRED-CLEAR BIT TO INDICATE
2617 012156 022626          CMP      (SP)+,(SP)+
2618 012160 000745          BR       3$
2619          ;4K BLOCK NOT PRESENT
2620          ;ADJUST STACK
2621          ;CHECK NEXT BLOCK
2621          ;NO KT PRESENT - MAP MAX OF 28K IN 4K CONTIGUOUS BLOCKS
2622 012162 012767 000001 166360  MAPMB:  MOV      #1,NOKT
2623 012170 022626          CMP      (SP)+,(SP)+
2624 012172 012737 012272 000004      MOV      #3,@#4
2625 012200 012767 000177 166656      MOV      #177,MEML
2626 012206 005067 166654          CLR      MEMH
2627 012212 012767 000001 166300      MOV      #1,BITPT
2628 012220 005001          CLR      #1,BITPT
2629 012222 005761 001000 1$:  TST     1000(1)
2630 012226 005761 005000          TST     5000(1)
2631 012232 005761 011000          TST     11000(1)
2632 012236 005761 015000          TST     15000(1)
2633 012242 082701 020000 2$:  ADD     #20000,R1
2634 012246 063657 166246          ASL     BITPT
2635 012252 022767 000200 166240      CMP      #200,BITPT
2636 012260 003360          BGT     1$
2637 012262 012737 000006 000004      MOV      #6,@#4
2638 012270 000207          RTS
2639 012272 046767 166222 166564 3$:  BIC      BITPT,MEML
2640          ;TIMEOUT OCCURRED- CLEAR BIT TO
2641 012300 022626          CMP      (SP)+,(SP)+
2642 012302 000757          BR       2$
2643          ;INDICATE 4K BLOCK NOT PRESENT
2644          ;ADJUST STACK POINTER
2645          ;*****
2646          ;MAP PARITY CORE AND CORRESPONDENCE TO ASSOCIATED REGISTERS
2647          ;*****
2648 012304 013767 001064 166234  MAPREG:  MOV      @#MEML,MEMUT
2649 012312 012767 000001 166200      MOV      #1,BITPT
2650 012320 012767 000001 166700      MOV      #1,KTSTART
2651 012326 005067 166232          CLR     RELOC
2652 012332 005067 166206          CLR     TBANK
2653 012336 005067 166210          CLR     HIWORD
2654          ;INITIALIZE ADDRESS OF TEST BANK
2655 012342 005067 166662          CLR     ADRTYP
2656          ;BEING CHECKED
2657          ;SET WRITE WRONG PARITY IN ALL REGISTERS PRESENT
2658          ;THEN WRITE TEST LOCATION VIA DATO AND READ TEST LOCATION VIA DATI
2659          ;THEN CLEAR WRITE WRONG PARITY IN ALL REGISTERS
2660 012346 012702 000566  MAPRB:  MOV      #MPRO,R2
2661 012352 032712 000001 1$:  BIT     #1,(2)
2662 012356 001003          BNE     +10
2663 012360 012772 000004 000000      MOV      #WWP,@(2)
2664          ;LOAD ADDRESS OF TABLE
2665          ;IS THIS REGISTER PRESENT?
2666          ;NO, GET NEXT ONE
2667          ;YES, SET WRITE WRONG PARITY AND CLEAR REST
    
```

```

2664 012366 062702 000010          ADD     #10,R2
2665 012372 020227 000766          CMP     R2,#TREG
2666 012376 103765          BLO     1$
2667 012400 016703 166140          MOV     TBANK,R3
2668 012404 066703 166620          ADD     ADRTYP,R3
2669 012410 011313          MOV     (3),(3)
2670 012412 005713          TST     (3)
2671 012414 012702 000566          MOV     #MPRO,R2
2672 012420 032712 000001 2$:  BIT     #1,(2)
2673 012424 001003          BNE     +10
2674 012426 042772 000004 000000      BIC     #WWP,@(2)
2675          ;CLEAR WRITE WRONG PARITY IN ALL
2676          ;PARITY REGISTERS
2676 012434 062702 000010          ADD     #10,R2
2677 012440 020227 000766          CMP     R2,#TREG
2678 012444 103765          BLO     2$
2679 012446 012702 000566  MAPRC:  MOV     #MPRO-10,R2
2680 012452 052702 000010 1$:  ADD     #10,R2
2681 012456 020227 000766          CMP     R2,#TREG
2682 012462 002031          BGE     MAPRD
2683 012464 032712 000001          BIT     #1,(2)
2684 012470 001370          BNE     1$
2685 012472 005772 000000          TST     @(R2)
2686          ;BRANCH IF DONE WITH TABLE
2687          ;IS THIS REGISTER PRESENT?
2688          ;NO, CHECK NEXT
2689          ;YES, WHICH 64K IS UNDER TEST?
2690 012500 005767 166046          BPL     1$
2691 012504 001004          TST     HIWORD
2692 012506 056762 166006 000002      BNE     2$
2693 012514 004033          BLS     BITPT,2(2)
2694 012516 056762 165776 000004 2$:  BR      3$
2695 012524 105762 000007 3$:  BIS     BITPT,4(2)
2696          ;SET BIT IN MAP FOR THIS PARITY REGISTER
2697          ;IS THIS THE FIRST ADDRESS FOUND FOR
2698          ;THIS PARITY REGISTER?
2699          ;NO, BRANCH
2700 012530 001005          BNE     4$
2701 012532 116762 166472 000006      MOV     ADRTYP,6(2)
2702 012540 105262 000007          INCB   7(2)
2703          ;INDICATE AN ADDRESS HAS BEEN FOUND FOR
2704          ;THIS REGISTER
2705          ;CLEAR BAD PARITY
2706 012544 000742          BR      1$
2707 012546 011313 166452  MAPRD:  MOV     @R3,@R3
2708 012550 062767 000002          ADD     #2,ADRTYP
2709 012554 026727 166446 000010      CMP     ADRTYP,#10
2710 012560 001402          BEQ     1$
2711 012562 000167 177554          JMP     MAPRB
2712 012564 005767 165754          TST     HIWORD
2713 012566 001021          BNE     MAPRE
2714 012568 001021          CMP     BITPT,#100
2715 012570 026727 165714 000100      BHS     MAPRE
2716 012572 103015          ADD     #20000,TBANK
2717 012574 062767 020000 165726      ASL     BITPT
2718 012576 006367 165676          CLR     ADRTYP
2719 012578 005067 166402          BIT     BITPT,MEMUT
2720 012580 036767 165666          BEQ     1$
2721 012582 001756          JMP     MAPRB
2722 012584 000167 177504          TST     NOKT
2723 012586 005767 165702          BEQ     +4
2724 012588 001401          RTS
2725 012590 000207          TST     KTSTART
2726 012592 005767 166350          BEQ     1$
2727 012594 001417          ;YES, BRANCH
    
```

```

2720 012660 012767 020000 165656 MOV #20000,TBANK ;NO, INIT TBANK TO SELECT KERNEL PAGE 1
2721 012665 012767 001400 165670 MOV #1400,RELOC ;SET UP FOR ACCESS TO 2BK BANK
2722 012674 004767 001012 JSR %7,NRALL ;INITIALLY MAP ALL PAGES NR, BANK 0
2723 012700 004767 001156 JSR %7,MAP1 ;MAP KERNEL 0 TO BANK 0, RW
2724 ;MAP KERNEL 7 TO THE EXTERNAL BANK, RW
2725 ;SET KERNEL 1 RW AND TURN ON KT11
2726 012704 005067 166316 CLR KTSTART ;INDICATE KT11 NOW IN USE
2727 012710 012737 000001 177572 MOV #1,@#SRO ;TURN ON KT11
2728 012716 006367 165576 1$: ASL BITPT ;SHIFT BANK INDICATOR
2729 012722 103011 BCC 2$ ;BRANCH IF FIRST 64K NOT DONE
2730 012724 012767 000001 165566 MOV #1,BITPT ;IF FIRST 64K DONE, SETUP FOR
2731 012732 013767 001066 165606 MOV @#MEMH, MEMUT ;SECOND 64K
2732 012740 012767 000001 165604 MOV #1,HIWORD ;INDICATE NOW TESTING HIGH 64K
2733 012746 062767 000200 165610 2$: ADD #200,RELOC
2734 012754 022767 007600 165602 CMP #7600,RELOC ;UP TO EXTERNAL BANK YET?
2735 012762 003003 BGT 3$ ;NO, CONTINUE
2736 012764 005037 177572 CLR @#SRO ;YES, TURN OFF KT11 AND EXIT
2737 012770 000207 RTS %7
2738 012772 036767 165522 165546 3$: BIT BITPT,MEMUT ;IS THIS 4K PRESENT?
2739 013000 001746 BEQ 1$ ;NO- BRANCH
2740 013002 016777 165556 166242 MOV RELOC,@KPAR1 ;YES, MAP PAGE 1 TO THIS BANK
2741 013010 005067 166214 CLR ADRTYP
2742 013014 000167 177326 JMP MAPRB ;GO TEST FOR PARITY MEMORY
2743
2744
2745
2746 ;*****
2747 ;ROUTINE TO TYPE MAP OF WHERE PARITY MEMORY IS PRESENT
2748 ;AND WHICH CONTROL REGISTERS CONTROL WHICH MEMORY
2749 ;*****
2750 013020 004767 000766 TMAP: JSR %7,CLRPAR ;CLEAR ALL PARITY REGISTERS PRESENT
2751 013024 104000 TYPE ;TYPE "THE PARITY REGISTERS CONTROL MEMORY
2752 013026 017074 MTMAP ;AS FOLLOWS:"
2753 013030 012701 000556 MOV #MPRO-10,R1 ;SET UP POINTER
2754 013034 062701 000010 TMAPA: ADD #10,R1
2755 013040 020127 000766 CMP R1,#TREG ;DONE WITH MAP?
2756 013044 002136 BGE TMAPEX ;YES, BRANCH
2757 013046 005067 165500 CLR HIWORD
2758 013052 005067 165444 CLR TRFLG ;INITIALIZE TRANSITION FLAG (USED TO
2759 ;FIGURE MEMORY LIMITS)
2760 013056 005067 165442 CLR TYFLG ;INITIALIZE TO INDICATE NOTHING TYPED FOR
2761 ;THIS REGISTER YET
2762 013062 012767 177774 165436 MOV #-4,TYCOR
2763 013070 016167 000002 165420 MOV 2(1),ADRPT ;GET LOW 64K MEMORY MAP WORD
2764 013076 012767 000001 165414 MOV #1,BITPT ;INITIALIZE 4K POINTER
2765 013104 032711 000001 BIT #1,(1) ;DOES THIS CONTROL EXIST?
2766 013110 001351 BNE TMAPA ;NO, GET ADDRESS OF NEXT ONE
2767 013112 011167 165376 MOV (1),TEMPX ;YES, PRINT ITS ADDRESS
2768 013116 094567 002274 JSR RS,OACNV
2769 013122 000514 TEMPX
2770 013124 017244 MPRAD
2771 013126 000006 6
2772 013130 104000 TYPE
2773 013132 017447 MX1
2774 013134 104000 TYPE
2775 013136 017244 MPRAD
    
```

```

2776 013140 062767 000004 165360 TMAPB: ADD #4,TYCOR ;KEEP TRACK OF # OF K OF CORE
2777 013146 036767 165346 165342 BIT BITPT,ADRPT ;DOES THIS PARITY REGISTER CONTROL THIS 4K?
2778 013154 001424 BEQ TMAPC ;NO- BRANCH
2779 013156 005767 165340 TST TRFLG ;YES, DOES IT CONTROL PREVIOUS 4K?
2780 013162 001043 BNE TMAPD ;YES- DON'T TYPE IT
2781 013164 012767 000001 165330 MOV #1,TRFLG ;NO- SET FLAG INDICATING TRANSITION
2782 013172 004567 002326 JSR RS,BDCNV ;CONVERT K CORE TO ASCII
2783 013176 000526 TYCOR
2784 013200 017144 MTYCOR
2785 013202 000003 3
2786 013204 104000 TYPE ;TYPE "CONTROLS", AND ADDRESS OF CORE
2787 013206 017546 MX2
2788 013210 104000 TYPE
2789 013212 017144 MTYCOR
2790 013214 104000 TYPE
2791 013216 017151 MDASH
2792 013220 005267 165300 INC TYFLG ;INDICATE TYPED
2793 013224 000422 BR TMAPD
2794 013226 005767 165270 TMAPC: TST TRFLG ;DID THIS PARITY REGISTER CONTROL PREVIOUS 4K?
2795 013232 001417 BEQ TMAPD ;NO, SKIP PRINTING
2796 013234 005067 165262 CLR TRFLG ;YES, TRANSITION OCCURRED- CLEAR FLAG
2797 013240 004567 002260 JSR RS,BDCNV ;CONVERT K CORE TO ASCII
2798 013244 000526 TYCOR
2799 013246 017144 MTYCOR
2800 013250 000003 3
2801 013252 104000 TYPE
2802 013254 017144 MTYCOR ;TYPE RIGHT AND RETURN
2803 013256 104000 TYPE
2804 013260 017157 MK
2805 013262 104000 TYPE
2806 013264 017154 MCR
2807 013266 005267 165232 INC TYFLG ;INDICATE TYPED
2808 013272 006367 165222 TMAPD: ASL BITPT ;UPDATE BIT POINTER TO NEXT 4K
2809 013276 103320 BCC TMAPB ;TEST NEXT 4K IF NOT DONE WITH 1ST 64K
2810 013300 005767 165246 TST HIWORD ;64-124K DONE?
2811 013304 001405 BEQ 1$ ;NO, BRANCH
2812 013306 005767 165212 TST TYFLG ;YES, WAS ANY PARITY MEMORY
2813 ;FOUND FOR THIS REGISTER?
2814
2815 013312 001250 BNE TMAPA
2816 013314 104002 ERROR ;NO PARITY MEMORY WAS FOUND FOR THIS
2817 ;REGISTER- EITHER WRITE WRONG PARITY
2818 ;FAILED, PARITY ERROR GENERATE OR
2819 ;DETECT FAILED, OR THE PARITY ERROR
2820 ;BIT FAILED TO SET
2821 013316 000646 BR TMAPA
2822 013320 016167 000004 165170 1$: MOV 4(1),ADRPT ;UPDATE TO MAP WORD FOR THE UPPER 64K
2823 013326 012767 000001 165164 MOV #1,BITPT ;RESET BIT POINTER
2824 013334 005267 165212 INC HIWRD ;INDICATE LOW 64K DONE
2825 013340 000677 BR TMAPB ;LOOP
2826 013342 000207 TMAPEX: RTS %7 ;RETURN WHEN DONE
2827
2828 ;ROUTINE TO HANDLE 8K SYSTEMS
2829 ;
2830 ;
2831 013344 012767 000002 165146 SMLSYS: MOV #2,BITPT ;SET UP POINTER FOR BANK 1
    
```

```

2832 013352 005037 000042 CLR @#42 ;MAKE EXIT STANDALONE
2833 013356 104000 TYPE
2834 013360 017466 NQMON
2835 013362 000207 RTS PC
2836
2837
2838
2839 ;*****
2840 ;ERROR HANDLER
2841 ;*****
2842 ;ERRORS CALL ENTERS HERE
2843 ;TYPES PC, ICNT, MPR ADDRESS, AND MPR CONTENTS
2844 ;TREG SHOULD CONTAIN ADDRESS OF PARITY REGISTER
2844 013364 012767 016517 000266 ERRST: MOV #MSTR,ERRB ;SETUP TO TYPE MPR ADDRESS AND CONTENTS
2845 013372 012767 177777 000262 MOV #-1,ERRBX
2846 013400 012767 000240 000256 MOV #240,ERRBX+2 ;NOP LOCATION AFTER MESSAGE
2847 013406 017767 165354 165124 MOV @TREG,TRDATA ;SETUP DATA
2848 013414 004567 001776 JSR R5,OACNV ;CONVERT TO ASCII
2849 013420 000766 TREG
2850 013422 016526 MTREG
2851 013424 000006 G
2852 013426 004567 001764 JSR R5,OACNV ;CONVERT TO ASCII
2853 013432 000540 TRDATA
2854 013434 016550 MDATA
2855 013436 000006 G
2856 013440 000461 BR ERRA
2857
2858
2859 ;ERRORP CALL ENTERS HERE
2860 ;TYPES PC, ICNT, MPR ADDRESS, MPR CONTENTS
2861 ;TEST LOCATION ADDRESS, VALUE EXPECTED, VALUE FOUND
2862 ;R5 MUST CONTAIN ADDRESS OF TEST LOCATION
2863 ;SHDBE MUST CONTAIN EXPECTED VALUE
2864 ;WAS MUST CONTAIN ACTUAL DATA
2865 ;TREG MUST CONTAIN ADDRESS OF PARITY REGISTER
2866 013442 012767 016517 000210 ERRP: MOV #MSTR,ERRB ;IN ADDITION TO BASIC PRINTOUT, TYPE
2867 ;MPR ADDRESS AND CONTENTS
2868 013450 012767 016561 000204 MOV #MSTRX,ERRBX ;ALSO OUTPUT DATA EXPECTED AND ACTUAL
2869 013456 012767 177777 000200 MOV #-1,ERRBX+2 ;NOP LOCATION AFTER MESSAGE
2870 013464 010567 165042 MOV R5,TSTLOC ;STORE ADDRESS BEING TESTED
2871 013470 017767 165272 165042 MOV @TREG,TRDATA
2872 013476 004567 001714 JSR R5,OACNV
2873 013502 000766 TREG
2874 013504 016526 MTREG
2875 013506 000006 G
2876 013510 004567 001702 JSR R5,OACNV
2877 013514 000540 TRDATA
2878 013516 016550 MDATA
2879 013520 000006 G
2880 013522 004567 001670 JSR R5,OACNV
2881 013526 000532 TSTLOC
2882 013530 016605 MSTRX1
2883 013532 000006 G
2884 013534 004567 001656 JSR R5,OACNV
2885 013540 000534 SHDBE
2886 013542 016621 MSTRX3
2887 013544 000006 G
    
```

```

2888 013546 004567 001644 JSR R5,OACNV
2889 013552 000536 WAS
2890 013554 016635 MSTRX5
2891 013556 000006 G
2892 013560 000411 BR ERRA
2893
2894
2895
2896 ;ERROR CALL ENTERS HERE
2897 ;TYPE PC AND ICNT ONLY
2898 013562 012767 177777 000070 ERR: MOV #-1,ERRB ;SET UP ONE MESSAGE CALL
2899 013570 012767 000240 000064 MOV #240,ERRBX
2900 013576 012767 000240 000060 MOV #240,ERRBX+2
2901 013604 032777 020000 165266 ERRA: BIT #BIT13,@SWR ;INHIBIT ERROR PRINT?
2902 013612 001025 BNE ERRC ;YES- BRANCH
2903 013614 011657 000070 MOV (SP),ERRD ;NO- DEVELOP CALLING ADDRESS
2904 013620 162767 000002 000062 SUB #2,ERRD
2905 013626 004567 001564 JSR R5,OACNV ;GO TO OCTAL TO ASCII CONVERT
2906 013632 013710 ERRA1: ERRD ;SOURCE ADDRESS
2907 013634 016472 ERRA2: MPC ;DESTINATION ADDRESS
2908 013636 000006 G ;#OF DIGITS TO CONVERT
2909 013640 004567 001552 JSR R5,OACNV ;CONVERT ICNT TO ASCII
2910 013644 016460 ERRA3: ICNT
2911 013646 016510 ERRA4: MICNT
2912 013650 000006 G
2913 013652 004567 001506 JSR R5,TYPSX ;TYPE MESSAGE
2914 013656 016464 ERRA5: MEO ;ERROR HEADER
2915 013660 000000 ERRB: OPEN ;ADDITIONAL ERROR MESSAGES IF ANY
2916 013662 000000 ERRBX: OPEN
2917 013664 177777 -1
2918 013666 023737 000042 000046 ERRC: CMP @#42,@#46 ;ARE WE IN ACT11 AUTOMATIC MODE?
2919 013674 001403 BEQ .+10 ;YES, HALT ON ERROR
2920 013676 005777 165176 TST @SWR ;HALT ON ERROR SET?
2921 013702 100001 BPL .+4 ;NO- BRANCH
2922 013704 000000 HALT ;YES- ERROR OCCURRED SO HALT
2923 013706 000002 RTI
2924 013710 000000 ERRD: OPEN
2925
2926
2927
2928 ;MAP ALL PAGES NON-RESIDENT, BANK 0
2929 013712 013746 000004 NRALL: MOV @#4,-(SP)
2930 013716 013746 000006 MOV @#6,-(SP)
2931 013722 012737 000006 000004 MOV #6,@#4
2932 013730 012737 000002 000006 MOV #RTI,@#6
2933 013736 010146 MOV R1,-(SP)
2934 013740 010246 MOV R2,-(SP)
2935 013742 010346 MOV R3,-(SP)
2936 013744 012701 001232 MOV #PDRTAB,R1
2937 013750 012703 000040 1$: MOV #32,R3
2938 013754 012102 MOV (R1)+,R2
2939 013756 005022 2$: CLR (R2)+
2940 013760 005303 DEC R3
2941 013762 001375 BNE 2$
2942 013764 020127 001236 CMP R1,#PDREND
2943 013770 003767 BLE 1$
    
```

```

2944 013772 012603      MOV      (SP)+,R3
2945 013774 012602      MOV      (SP)+,R2
2946 013776 012501      MOV      (SP)+,R1
2947 014000 012637 000006  MOV      (SP)+,@#6
2948 014004 012637 000004  MOV      (SP)+,@#4
2949 014010 000207      RTS      %7
2950
2951
2952
2953
2954 014012 010146      ;ROUTINE TO CLEAR ALL PARITY REGISTERS PRESENT
CLRPAR: MOV      R1,-(SP)
2955 014014 010246      MOV      R2,-(SP)
2956 014016 010701      MOV      PC,R1
2957 014020 062701 164546  ADD      #MPRO--,R1
2958 014024 010702      MOV      PC,R2
2959 014026 062702 164740  ADD      #TREG--,R2
2960 014032 032711 000001  1$: BIT      #1,@R1      ;IS THIS REGISTER PRESENT?
2961 014036 001002      BNE      +6
2962 014040 005071 000000  CLR      @(R1)      ;CLEAR ALL PARITY REGISTERS
2963 014044 062701 000010  ADD      #10,R1
2964 014050 020102      CMP      R1,R2
2965 014052 103767      BLO     1$
2966 014054 012602      MOV      (SP)+,R2
2967 014056 012601      MOV      (SP)+,R1
2968 014060 000207      RTS      %7
2969
2970
2971
2972
2973
2974
2975 014062 012777 077406 165150  MAP1: MOV      #77406,@KPDR0
2976 014070 012777 077406 165144  MOV      #77406,@KPDR1
2977 014076 012777 077406 165142  MOV      #77406,@KPDR7
2978 014104 012777 007600 165144  MOV      #7600,@KPAR7
2979 014112 005077 165132      CLR      @KPAR0
2980 014116 012737 000001 177572  MOV      #1,@#SR0
2981 014124 000207      RTS      %7
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991 014126 010246      ;ROUTINE TO LOCATE THE FIRST PARITY MEMORY ADDRESS (ABOVE BANK 0)
2992 014130 012702 000200  ;CORRESPONDING TO A GIVEN PARITY REGISTER-REQUIRES THAT THE ROUTINES
2993 014134 012701 000002  ;MAPREM AND MAPREG HAVE ALREADY BEEN RUN
2994 014140 005767 164404  ;TO USE, PUT THE ADDRESS OF THE ADDRESS OF THE REGISTER IN R0 (I.E. POINT TO
2995 014144 001403      ;TO MAP TABLE)-THE DESIRED ADDRESS IS RETURNED IN R1, USING KERNEL PAGE 1 IF
2996 014146 022701 000200  ;KT11 IS PRESENT
2997 014152 014200      LOCATM: MOV     R2,-(SP)
2998
2999 014154 030160 000002  3$: BIT      R1,2(R0)      ;SKIP USE OF BANK 0
                                ;KT11 PRESENT?
                                ;YES, BRANCH
                                ;NO, CHECK ONLY FOR MEMORY IN FIRST 28K
                                ;IF NO MEMORY FOUND IN 1ST 28K, GIVE
                                ;ERROR RETURN
                                ;DOES THIS 4K CORRESPOND TO THIS REGISTER?

```

```

3000 014160 001021      BNE     LOCAT1      ;YES, BRANCH
3001 014162 062702 000200  ADD     #200,R2      ;NO, CHECK TO SEE IF NEXT 4K CORRESPONDS
3002 014166 006301      ASL     R1
3003 014170 103363      BCC     1$
3004 014172 012701 000001  MOV     #1,R1
3005 014176 030160 000004  2$: BIT     R1,4(R0)      ;CHECK HIGH 64K
3006 014202 001010      BNE     LOCAT1
3007 014204 062702 000200  ADD     #200,R2
3008 014210 006301      ASL     R1
3009 014212 103371      BCC     2$
3010 014214 012701 000001  4$: MOV     #1,R1      ;NO PARITY MEMORY CORRESPONDS TO
3011 014220 012602      MOV     (SP)+,R2      ;THIS REGISTER- RETURN WITH ERROR
3012 014222 000207      RTS     %7      ;INDICATION
3013 014224 005767 164320  LOCAT1: TST     NOKT      ;KT11 PRESENT?
3014 014230 001005      BNE     1$      ;NO- BRANCH OVER
3015 014232 010277 165014  MOV     R2,@KPAR1      ;YES- SETUP R1 TO REFERENCE 4K
3016 014236 012701 020000  MOV     #20000,R1      ;BANK USING KERNEL PAGE 1
3017 014242 000407      BR      2$
3018 014244 010201  1$: MOV     R2,R1      ;SETUP R1 TO REFERENCE 4K BANK
3019 014246 006301      ASL     R1      ;WITHOUT KT11
3020 014250 006301      ASL     R1
3021 014252 006301      ASL     R1
3022 014254 006301      ASL     R1
3023 014256 006301      ASL     R1
3024 014260 006301      ASL     R1
3025 014262 116067 000006 000010  2$: MOV     6(R0),LSAV      ;RESTORE R2
3026 014270 066701 000004  ADD     LSAV,R1      ;AND RETURN
3027 014274 012602      MOV     (SP)+,R2
3028 014276 000207      RTS     %7
3029 014300 000000  LSAV:  0
3030
3031
3032
3033
3034 014302 013746 000004  ;SAVE LOADER IN TOP OF FIRST 4K
SAVLDR: MOV     @#4,-(SP)      ;SAVE CONTENTS OF TIMEOUT VECTOR
3035 014306 013746 000006  MOV     @#6,-(SP)
3036 014312 012737 000006 000004  MOV     #6,@#4
3037 014320 012737 000002 000006  MOV     #RTI,@#6      ;SETUP TO RTI ON TIMEOUT
3038 014326 010146  MOV     R1,-(SP)      ;SAVE REGISTERS
3039 014330 010246  MOV     R2,-(SP)
3040 014332 012701 152234  MOV     #152234,R1
3041 014336 000261  1$: SEC
3042 014340 005711      TST     @R1      ;IF TIMEOUT, C BIT WILL STILL BE SET
3043 014342 103006      BCC     2$      ;IF NO TIMEOUT, C BIT WILL BE CLEAR
3044 014344 162701 020000  SUB     #20000,R1      ;TIMEOUT OCCURRED, CHECK FOR NEXT LOWER BANK
3045 014350 020127 020000  CMP     R1,#20000
3046 014354 101370      BHI     1$
3047 014356 000410      BR      SAVLDX
3048 014360 012702 032234  2$: MOV     #32234,R2      ;ONLY 4K OF 1ST 28K PRESENT- EXIT
3049 014364 012122  3$: MOV     (R1)+,(R2)+      ;THIS BANK IS HIGHEST OF 1ST 28K- COPY
3050 014366 020227 040000  CMP     R2,#40000      ;LOADER AREA TO BANK 0
3051 014372 103774      BLO     3$
3052 014374 005267 000016  INC     LDRSVD      ;INDICATE LOADER HAS BEEN MOVED TO
3053
3054 014400 012602  SAVLDX: MOV     (SP)+,R2      ;THE TOP OF THE FIRST 4K
3055 014402 012601      MOV     (SP)+,R1

```

```

3056 014404 012637 000006      MOV      (SP)+,@#6
3057 014410 012637 000004      MOV      (SP)+,@#4
3058 014414 000207              RTS      %7
3059 014416 000000      LDRSVD: 0
3060
3061
3062
3063      ;ROUTINE TO RESTORE THE LOADER FROM BANK 0 (WHERE IT WAS SAVED) TO THE
3064      ;HIGHEST BANK IN THE FIRST 28K OF MEMORY
3065 014420 005767 177772      RSTLDR: TST  LDRSVD
3066 014424 001431              BEQ      RSTLDX
3067 014426 012737 000006 000004      MOV      #6,@#4
3068 014434 012737 000002 000006      MOV      #RTI,@#6
3069 014442 012701 152234      MOV      #152234,R1
3070 014446 000261      1$: SEC
3071 014450 005711              TST      @R1
3072 014452 103006              BCC      2$
3073 014454 162701 020000              SUB      #20000,R1
3074 014460 020127 020000              CMP      R1,#20000
3075 014464 101370              BHI      1$
3076 014466 000410              BR       RSTLDX
3077 014470 012702 032234      2$: MOV      #32234,R2
3078 014474 012221      3$: MOV      (R2)+,(R1)+
3079 014476 020227 040000              CMP      R2,#40000
3080 014502 103774              BLO      3$
3081 014504 104000              TYPE
3082 014506 017560      LDRMSG
3083 014510 000207      RSTLDR: RTS  PC
3084 014512 000776      BR       -2
3085
3086
3087
3088      ;SCAN ALL MEMORY FOR BAD PARITY, TYPE 18 BIT ADDRESSES OF
3089      ;LOCATIONS FOUND TO BE BAD, AND WRITE INTO LOCATIONS WITH GOOD PARITY
3090 014514 010146      PSCAN: MOV  R1,-(SP)
3091 014516 010246              MOV      R2,-(SP)
3092 014520 010346              MOV      R3,-(SP)
3093 014522 010446              MOV      R4,-(SP)
3094 014524 013746 000004      MOV      @#4,-(SP)
3095 014530 013746 000006      MOV      @#6,-(SP)
3096 014534 013746 000114      MOV      @#114,-(SP)
3097 014540 013746 000116      MOV      @#116,-(SP)
3098 014544 012737 000006 000004      MOV      #6,@#4
3099 014552 005037 000006      CLR      @#6
3100 014556 012737 000116 000114      MOV      #116,@#114
3101 014564 005037 000116      CLR      @#116
3102 014570 005767 163754      TST      NOKT
3103 014574 001513              BEQ      PSCAN1
3104 014576 005002              CLR      R2
3105 014600 012703 000001      MOV      #1,R3
3106 014604 004767 177202      1$: JSR      %7,CLRPAR
3107 014610 005712              TST      @R2
3108 014612 000240              NOP
3109 014614 012701 000566      MOV      #MPRO,R1
3110
3111 014620 032711 000001      3$: BIT      #1,@R1
    
```

```

3112 014624 001003              BNE      .+10
3113 014626 005771 000000      TST      @R1
3114 014632 100424              BMI      6$
3115 014634 062701 000010      ADD      #10,R1
3116 014640 020127 000766      CMP      R1,#TREG
3117 014644 103765              BLO      3$
3118 014646 062702 000002      4$: ADD      #2,R2
3119 014652 032702 017777      BIT      #17777,R2
3120 014656 001352              BNE      1$
3121 014660 006303              5$: ASL      R3
3122 014662 020327 000200      CMP      R3,#200
3123 014666 103035              BHS      PSCANX
3124 014670 030367 164170      BIT      R3,MEML
3125 014674 001343              BNE      1$
3126 014676 062702 020000      ADD      #20000,R2
3127 014702 000766              BR       5$
3128 014704 010267 000110      6$: MOV      R2,PSADRS
3129 014710 004567 000502      JSR      R5,DACNV
3130 014714 015020      PSADRS
3131 014716 017440      MPSPER1
3132 014720 000006              6
3133 014722 104000              TYPE
3134 014724 017410      MPSPER
3135 014726 032777 002000 164144      BIT      #2000,@SWR
3136 014734 001401              BEQ      .+4
3137 014736 000000              HALT
3138 014740 011212              MOV      @R2,@R2
3139
3140 014742 005071 000000      CLR      @R1
3141 014746 005712              TST      @R2
3142 014750 005771 000000      TST      @R1
3143 014754 100001              BPL      .+4
3144 014756 104002      ERROR
3145
3146 014760 000732              BR       4$
3147 014762 004767 177024      PSCANX: JSR  PC,CLRPAR
3148 014766 012637 000116      MOV      (SP)+,@#116
3149 014772 012637 000114      MOV      (SP)+,@#114
3150 014776 012637 000006      MOV      (SP)+,@#6
3151 015002 012637 000004      MOV      (SP)+,@#4
3152 015006 012604              MOV      (SP)+,R4
3153 015010 012603              MOV      (SP)+,R3
3154 015012 012602              MOV      (SP)+,R2
3155 015014 012601              MOV      (SP)+,R1
3156 015016 000207              RTS      %7
3157 015020 000000      PSADRS: 0
3158 015022 000000      PSCANH: 0
3159
3160
3161
3162      ;SCAN ALL MEMORY FOR BAD PARITY USING KT11
3163      ;TYPE 18 BIT ADDRESSES OF LOCATIONS FOUND BAD, AND WRITE GOOD PARITY BACK IN
3164 015024 017746 164222      PSCAN1: MOV  @KPAR1,-(SP)
3165 015030 032737 000001 177572      BIT      #1,@MSRO
3166 015036 001004              BNE      1$
3167 015040 004767 176646      JSR      PC,NRALL
    
```



```

3168 015044 004767 177012          JSR      PC,MAP1          ;MAP KERNEL 7 TO THE EXTERNAL BANK, RW
3169                                     ;MAP KERNEL 1 RW, AND TURN ON KT11
3170 015050 005067 177746          1$: CLR      PSCANH          ;CLEAR FLAG TO INDICATE CHECKING FIRST 64K
3171 015054 005077 164172          CLR      @KPAR1          ;INITIALIZE TO BANK 0
3172 015060 012703 000001          PSLOOP: MOV     #1,R3          ;R3 IS USED AS A BIT POINTER
3173 015064 005767 177732          PSLUP: TST     PSCANH          ;TESTING TOP 64K?
3174 015070 001004 000000          BNE     2$                ;YES, BRANCH
3175 015072 030367 163766          BIT     R3,MEML          ;NO, IS PARITY MEMORY PRESENT IN THIS 4K?
3176 015076 001022 000000          BNE     PSXTST          ;YES- GO TEST IT
3177 015100 000403 000000          BR      PSNXT          ;NO- CHECK FOR NEXT 4K
3178 015102 030367 163760          2$: BIT     R3,MEMH          ;IS PARITY MEMORY PRESENT IN THIS 4K?
3179 015106 001016 000000          BNE     PSXTST          ;YES- GO TEST IT
3180 015110 062777 000200 164134 PSNXT: ADD     #200,@KPAR1      ;NO- MAP TO NEXT 4K
3181 015116 006303 000000          ASL     R3                ;BRANCH IF NOT END OF 64K
3182 015120 103361 000000          BCC     PSLUP          ;END OF TOP 64K?
3183 015122 005767 177674          TST     PSCANH          ;YES, GET READY TO EXIT
3184 015126 001003 000000          BNE     PSCX1          ;NO, SET FLAG INDICATING DONE WITH
3185 015130 005267 177666          INC     PSCANH          ;LOWER 64K
3186                                     ;
3187 015134 000751 000000          BR      PSLOOP          ;
3188 015136 012677 164110          PSCX1: MOV     (SP)+,@KPAR1    ;
3189 015142 000707 000000          BR      PSCANX          ;
3190 015144 012702 020000          PSXTST: MOV    #2000,R2        ;R2 USED AS ADDRESS POINTER
3191 015150 004767 176636          1$: JSR      %7,CLRPAR        ;CLEAR ALL PARITY REGISTERS
3192 015154 005712 000000          TST     @R2              ;READ LOCATION
3193 015156 012701 000566          MOV     #MPRO,R1         ;SETUP TO SCAN REGISTERS FOR PARITY ERROR SET
3194 015162 032711 000001          2$: BIT     #1,@R1          ;
3195 015166 001003 000000          BNE     .+10            ;
3196 015170 005771 000000          TST     @(R1)           ;PARITY ERROR SET?
3197 015174 100413 000000          BMI     4$              ;YES, BRANCH
3198 015176 062701 000010          ADD     #10,R1          ;NO, CHECK NEXT
3199 015202 020127 000766          CMP     R1,#TREG        ;
3200 015206 103765 000000          BLD     2$              ;LOOP UNTIL ALL REGISTERS HAVE BEEN CHECKED
3201 015210 062702 000002          ADD     #2,R2           ;UPDATE TEST ADDRESS POINTER
3202 015214 020227 040000          CMP     R2,#40000       ;DONE WITH BANK?
3203 015220 103753 000000          BLO     1$              ;NO- LOOP
3204 015222 000732 000000          BR      PSNXT          ;YES- GO CHECK FOR ANOTHER BANK
3205 015224 010267 177570          4$: MOV     R2,PSADRS      ;PARITY ERROR OCCURRED- GET 18 BIT
3206 015230 042767 160000 177562 BIC     #160000,PSADRS    ;OCTAL ADDRESS OF BAD LOCATION
3207 015236 035046 000000          CLR     -(SP)           ;
3208 015240 017746 164006          MOV     @KPAR1,-(SP)     ;
3209 015244 006316 000000          ASL     @SP              ;
3210 015246 006316 000000          ASL     @SP              ;
3211 015250 006316 000000          ASL     @SP              ;
3212 015252 006316 000000          ASL     @SP              ;
3213 015254 006316 000000          ASL     @SP              ;
3214 015256 006166 000002          ROL     2(SP)           ;
3215 015262 006316 000000          ASL     @SP              ;
3216 015264 006166 000002          ROL     2(SP)           ;
3217 015270 006366 000002          ASL     2(SP)           ;
3218 015274 062667 177520          ADD     (SP)+,PSADRS    ;
3219 015300 004567 000112          JSR     R5,OACNV        ;CONVERT LOW 16 OCTAL BITS TO ASCII
3220 015304 015020 000000          PSADRS MPSE1            ;
3221 015306 017440 000000          MPSE1  6                ;
3222 015310 000006 000000          6                ;
3223 015312 116704 002122          MOVB   MPSE1,R4        ;
    
```

```

3224 015316 062604 000000          ADD     (SP)+,R4          ;CHANGE TO ASCII FOR 18 BITS
3225 015320 110467 002114          MOVB   R4,MPSE1        ;
3226 015324 104000 000000          TYPE   MPSE1            ;TYPE ADDRESS OF LOCATION WITH BAD PARITY
3227 015326 017410 000000          MPSE1  17410           ;
3228 015330 032777 002000 163542 BIT     #2000,@SWR      ;SWITCH 10 SET?
3229 015336 001401 000000          BE^    .+4              ;NO- BRANCH
3230 015340 000000 000000          HALT                                ;HALT ON BAD PARITY SET
3231 015342 011212 000000          MOV     @R2,@R2          ;REWRITE LOCATION CONTAINING BAD PARITY
3232 015344 035071 000000          CLR     @(R1)           ;CLEAR PARITY ERROR BIT
3233 015350 005712 000000          TST     @R2              ;READ LOCATION TO SEE IF PARITY IS NOW GOOD
3234 015352 005771 000000          TST     @(R1)           ;CHECK PARITY ERROR BIT
3235 015356 100001 000000          BPL     .+4              ;
3236 015360 104002 000000          ERROR  3$              ;REWRITING LOCATION DID NOT CLEAR BAD PARITY
3237 015362 000712 000000          BR      3$              ;GO TEST NEXT LOCATION
3238                                     ;
3239                                     ;
3240                                     ;
3241                                     ;
3242                                     ;PIC ROUTINE TO OUTPUT A SERIES OF ASCII MESSAGES (CALLED VIA JSR R5)
3243 015364 012567 000022          TYP$X: MOV    (R5)+,TYP$BX    ;GET ADDRESS OF MESSAGE
3244 015370 022767 177777 000014 CMP     #-1,TYP$BX        ;TERMINATOR?
3245 015376 001001 000000          BNE     TYP$AX          ;NO, BRANCH
3246 015400 000205 000000          RTS     %5              ;YES, RETURN
3247 015402 013746 177776          TYP$AX: MOV    @#$PS,-(SP)    ;SETUP TO CALL TYPE ROUTINE VIA JSR
3248 015406 004767 163502          JSR     PC,$TYPE        ;TYPE ASCII MESSAGE
3249 015412 000000 000000          TYP$X: OPEN  BR      TYP$X          ;
3250 015414 000763 000000          BR      TYP$X          ;
3251                                     ;
3252                                     ;
3253                                     ;
3254                                     ;SUBROUTINE FOR OCTAL TO ASCII CONVERSION
3255 015416 013567 000074          OACNV: MOV    @(5)+,OACNVX    ;GET OCTAL VALUE
3256 015422 012567 000072          MOV     (5)+,OACDST        ;GET DESTINATION ADDRESS
3257 015426 012567 000070          MOV     (5)+,OACNT        ;GET CONVERT COUNT
3258 015432 066767 000064          ADD     OACNV,OACDST      ;DEVELOP ADDRESS TO STORE 1ST CHAR.
3259 015440 016746 000052          OACNVA: MOV   OACNVX,-(SP)    ;
3260 015444 042716 177770          BIC     #177770,@SP      ;ISOLATE LEAST SIGNIFICANT DIGIT
3261 015450 062716 000060          ADD     #60,@SP          ;CONVERT DIGIT TO ASCII
3262 015454 005367 000040          DEC     OACDST          ;
3263 015460 112677 000034          MOVB   (SP)+,@OACDST      ;STORE ASCII CHARACTER
3264 015464 042767 000007          BIC     #7,OACNVX        ;
3265 015472 006067 000020          ROR     OACNVX          ;
3266 015476 006067 000014          ROR     OACNVX          ;
3267 015502 006067 000010          ROR     OACNVX          ;
3268 015506 005367 000010          DEC     OACNT          ;DONE ALL DIGITS?
3269 015512 001352 000000          BNE     OACNVA          ;BRANCH IF NOT DONE
3270 015514 000205 000000          RTS     R5              ;DONE, EXIT
3271 015516 000000 000000          OACNVX: OPEN  ;
3272 015520 000000 000000          OACDST: 0              ;
3273 015522 000000 000000          OACNT: 0              ;
3274                                     ;
3275                                     ;
3276                                     ;
3277                                     ;SUBROUTINE FOR BINARY TO DECIMAL ASCII CONVERSION
3278 015524 104005 000000          BDCNV: SAV04            ;SAVE REGS
3279 015526 012700 015702          MOV     #DECVAL,%        ;SET UP ADDR TO STORE DECIMAL ASCII
    
```



```

3343 015766 103766          BLO 1$
3344 015770 010667 163264  MOV SP,SPSAV
3345 015774 000000          HALT
3346 015776 012737 015732 000024 PWRUP: MOV #PWRDN,@#24 ;POWER DOWN HALT
3347 016004 016706 163250          MOV SPSAV,SP ;SET UP FOR POWER DOWN
3348 016010 005027 000000          CL# #0 ;STALL SO OUTPUT WON'T BE GARBLED
3349 016014 005367 177772          DEC -2
3350 016020 104375          BNE -4
3351 016022 104000          TYPE ;TYPE RECOVERY MESSAGE
3352 016024 017344          MPWRF
3353 016026 000167 163244          JMP RSTART ;RESTART
3354
3355
3356
3357
;EMT HANDLER
3358 016032 011646          EMTINT: MOV (SP),-(SP) ;GET SAVED PC
3359 016034 162716 000002          SUB #2,(SP) ;DECREMENT PC BY 2
3360 016040 017616 000000          MOV @(SP),(SP) ;GET CALL
3361 016044 121667 000050          CMPB (SP),EMTLIM ;CHECK IF CALL WITHIN LIMITS
3362 016050 101402          BLOS EMTA
3363 016052 000000          HALT ;CALL IS NOT WITHIN LIMITS
3364 016054 000776          BR -2
3365 016056 006116          EMTA: ROL (SP) ;EMT ARG X 2
3366 016060 042716 177001          BIC #177001,(SP) ;REMOVE 7 MSB
3367 016064 062716 016076          ADD #EMTTAB,(SP) ;FORM EMT RTN ADDRESS
3368 016070 017616 000000          MOV @(SP),(SP)
3369 016074 000136          JMP @(SP)+ ;GO TO EMT RETURN
3370
3371
3372
3373 016076          ;EMT DEFINITIONS AND ASSIGNMENTS
EMTTAB:
3374          TYPE=EMT+EMTX
3375 016076 001114          $TYPE
3376          SCOPE=EMT+EMTX
3377 016100 016356          SCOPEC
3378          ERROR=EMT+EMTX
3379 016102 013562          ERR
3380          ERRORP=EMT+EMTX
3381 016104 013442          ERPP
3382          ERRORS=EMT+EMTX
3383 016106 013364          ERRST
3384          SAV04=EMT+EMTX
3385 016110 016122          SV04
3386          RST04=EMT+EMTX
3387 016112 014006          RS04
3388          RST05=EMT+EMTX
3389 016114 016236          RS05
3390          SAV05=EMT+EMTX
3391 016116 016142          SV05
3392 016120 000010          EMTLIM: EMTX-1
3393
3394
3395
;SUBROUTINE TO SAVE REGS 0-4
3396 016122 012666 177764          SV04: MOV (SP)+,-12.(SP) ;MOVE PC+PS UP STACK
3397 016126 012666 177764          MOV (SP)+,-12.(SP)
3398 016132 012767 000002 000040          MOV #RTI,SV05C
    
```

```

3399 016140 000411          BR SV05B
3400
3401
3402
3403
;SUBROUTINE TO SAVE REGS 0-5 + PLACE EMT PC IN R5
3404 016142 012767 000240 000030          SV05: MOV #NOP,SV05C
3405 016150 000400          BR SV05A
3406
;SUBROUTINE TO SAVE REGS 0-5
3407
SV05A: MOV (SP)+,-14.(SP)
3408 016152 012666 177762          MOV (SP)+,-14.(SP)
3409 016156 012666 177762          MOV R5,-(SP)
3410 016162 010546          SV05B: MOV R4,-(SP)
3411 016164 010446          MOV R3,-(SP)
3412 016166 010346          MOV R2,-(SP)
3413 016170 010246          MOV R1,-(SP)
3414 016172 010146          MOV %0,-(SP)
3415 016174 010046          CMP -(SP),-(SP)
3416 016176 024646          SV05C: RTI ;RTI OR NOP
3417 016200 000002          MOV 16.(SP),R5 ;EMT PC TO R5
3418 016202 016605 000020          RTI
3419 016206 000002
3420
3421
3422
;SUBROUTINE TO RESTORE REGS 0-4
3423
RS04: CMP (SP)+,(SP)+
3424 016210 022626          MOV (SP)+,%0
3425 016212 012600          MOV (SP)+,R1
3426 016214 012601          MOV (SP)+,R2
3427 016216 012602          MOV (SP)+,R3
3428 016220 012603          MOV (SP)+,R4
3429 016222 012604          MOV (SP)+,R5
3430 016224 016646 177764          MOV -12.(SP),-(SP) ;MOVE PC+PS DOWN STACK
3431 016230 016646 177764          MOV -12.(SP),-(SP)
3432 016234 000002          RTI
3433
3434
3435
;SUBROUTINE TO RESTORE REGS 0-5
3436
RS05S: MOV R5,16.(SP) ;SET EMT PC TO R5
3437 016236 010566 000020          CMP (SP)+,(SP)+
3438 016242 022626          MOV (SP)+,%0
3439 016244 012600          MOV (SP)+,R1
3440 016246 012601          MOV (SP)+,R2
3441 016250 012602          MOV (SP)+,R3
3442 016252 012603          MOV (SP)+,R4
3443 016254 012604          MOV (SP)+,R5
3444 016256 012605          MOV (SP)+,R6
3445 016260 016646 177762          MOV -14.(SP),-(SP)
3446 016264 016646 177762          MOV -14.(SP),-(SP)
3447 016270 000002          RTI
3448
3449
3450
;ROUTINE TO LOOP THRU A SINGLE INSTRUCTION TEST
3451          ;LOAD THE STARTING ADDRESS OF THE TEST
3452          ;YOU WISH TO RUN (THE ADDRESS OF THE TESTXX
3453          ;TAG) AT THE 1ST HALT, SET SWITCH REGISTER
3454
    
```

```

3455 ;OPTIONS AT THE 2ND HALT.
3456 ;NOTE THAT SW11 MUST BE DOWN AFTER THE 2ND HALT
3457 TESTX: CLR @#PS
3458 HALT ;WAIT FOR STARTING ADDRESS
3459 MOV @SWR,RETURN ;LOAD STARTING ADDRESS IN RETURN
3460 ADD #2,RETURN ;ADD 2 TO POINT TO INSTRUCTION AFTER
3461 HALT ;SET SR OPTIONS
3462 MOV #-1,TSTX ;SET FLAG
3463 BIT #10000,@SWR ;CHECK SW12
3464 BEQ .+12 ;BRANCH IF NOT SET
3465 BIC #20,@#PS ;CLEAR TRACE BIT
3466 BR .+10 ;SKIP NEXT INSTRUCTION
3467 BIS #20,@#PS ;SET TRACE BIT
3468 JMP @RETURN ;JUMP TO TEST
3469
3470
3471 ;SCOPE AND/OR ITERATION LOOP FOR EACH TEST 64 TIMES
3472 ;A SETUP ROUTINE SHOULD INITIALIZE RETURN AND IMAX
3473 SCOPE: BIT #40000,@SWR ;TEST SR FOR SCOPE
3474 BNE SCOPEB ;YES, SCOPE
3475 BIT #4000,@SWR ;NO-TEST FOR ITERATION
3476 BNE SCOPEG ;INHIBIT ITERATION
3477 TST PASCNT ;FIRST PASS?
3478 BEQ SCOPEG ;YES, INHIBIT ITERATIONS
3479 TST TSTX ;USING SINGLE SUBTEST STARTUP?
3480 BNE SCOPEB ;YES, LOOP
3481 CMP ICNT,IMAX ;COMPARE CURRENT COUNT TO MAX NUMBER
3482 BPL SCOPEG ;EXIT-DONE
3483 INC ICNT ;INCREMENT COUNT
3484 MOV (6)+,%6 ;REPOSITION STACK
3485 JMP @RETURN ;RESTORE PREVIOUS PROCESSOR STATUS
3486 SCOPEB: CLR TSTX ;REPEAT TEST
3487 CLR ICNT ;IF USING TESTX STARTUP, RETURN TO NORMAL FLOW
3488 MOV @%6,RETURN ;CLEAR COUNT
3489 RTI ;SAVE SCOPE RETURN POINTER
3490 IMAX: 100 ;RETURN INLINE-NEXT TEST
3491 ICNT: 0 ;ITERATION COUNT
3492 RETURN: 0 ;COUNT LOCATION FOR ITERATION LOOP
3493 ;ADDRESS OF LAST TEST
3494
3495
3496
3497
3498
3499 ;ASCII MESSAGES
3500 MEO:
3501 MTNUM: .ASCII <15><12>'PC= '
3502 MPC: .ASCII ' ICNT= '
3503
3504 MICNT: .ASCIZ ' '
3505 MSTR: .ASCII ' MPR= '
3506 MTREG: .ASCII ' MPR DATA= '
3507
3508
3509
3510
    
```

```

3511 MDATA: .ASCIZ ' '
3512 MSTRX: .ASCII <15><12>' TEST LOC= '
3513 MSTRX1: .ASCII ' '
3514 MSTRX3: .ASCII ' '
3515 MSTRX5: .ASCIZ ' '
3516 MSETSR: .ASCIZ <15><12>'SET SR OPTIONS'
3517 MCON: .ASCIZ ' , PRESS CONTINUE'
3518 MMDEV: .ASCIZ <15><12>'SET DEVICE ADDRESS IN SR'
3519 MMADR: .ASCIZ <15><12>'SET MEMORY TEST LOC IN SR'
3520 MMPAT: .ASCIZ <15><12>'SET TEST PATTERN IN SR'
3521 MPRS: .ASCIZ <15><12><12>'MEMORY PARITY REGISTERS PRESENT:'<15><12>'
3522 MMAP: .ASCIZ <15><12>'PARITY REGISTERS CONTROL MEMORY AS:'<15><12>'
3523 MTYCOR: .ASCIZ ' '
3524 MDASH: .ASCIZ '- '
3525 MCR: .ASCIZ <15><12>'
3526 MK: .ASCIZ 'K'
    
```

3567	017161	116	020117	040520	MT:	.ASCIZ	'NO PARITY MEMORY FOUND'<15><12>
3568	017166	044522	054524	046440			
3569	017174	046505	051117	020131			
3570	017202	047506	047125	006504			
3571	017210	000012					
3572	017212	047516	050040	051101	MTR:	.ASCIZ	'NO PARITY REGISTER FOUND'<15><12>
3573	017220	052111	020131	042522			
3574	017226	051507	042524	020122			
3575	017234	047506	047125	006504			
3576	017242	000012					
3577	017244	020040	020040	020040	MPRAD:	.ASCIZ	'<15><12>
3578	017252	020040	005015	000			
3579	017257	177	005015	005015	MTIT:	.ASCIZ	<177><15><12><15><12>'CCMFAFO MS11,MA11-P,MF11-LP PARITY MEMORY TESTS'
3580	017264	041503	043115	043101			
3581	017272	020060	051515	030461			
3582	017300	046454	030501	026461			
3583	017306	026120	043115	030461			
3584	017314	046055	020120	040520			
3585	017322	044522	054524	046440			
3586	017330	046505	051117	020131			
3587	017336	022524	052123	000123			
3588	017344	005015	047520	042527	MPWRF:	.ASCIZ	<15><12>'POWER FAILED'
3589	017352	020122	040506	046111			
3590	017360	042105	000				
3591	017363	007			MPGEND:	.BYTE	007
3592	017364	005015	047105	020104		.ASCII	<15><12>'END PASS = '
3593	017372	040520	051523	036440			
3594	017400	040					
3595	017401	040	020040	020040	MPCNT:	.ASCIZ	' '
3596	017406	000040					
3597	017410	006415	041012	042101	MPSER:	.ASCII	<15><15><12>'BAD PAR FOUND IN LOC '
3598	017416	050040	051101	043040			
3599	017424	052517	042116	044440			
3600	017432	020116	047514	020103			
3601	017440	020040	020040	020040	MPSER1:	.ASCIZ	' '
3602	017446	000					
3603	017447	015	051012	043505	MX1:	.ASCIZ	<15><12>'REGISTER AT '
3604	017454	051511	042524	020122			
3605	017462	052101	000040				
3606	017466	005015	047515	044516	NOMON:	.ASCIZ	<15><12>'MONITOR WILL NOT BE RESTORED FOR AN BK SYSTEM'
3607	017474	047524	020122	044527			
3608	017502	046114	047040	052117			
3609	017510	041040	020105	042522			
3610	017516	052123	051117	042105			
3611	017524	043040	051117	040440			
3612	017532	020116	045470	051440			
3613	017540	051531	042524	000115			
3614	017546	047503	052116	047522	MX2:	.ASCIZ	'CONTROLS '
3615	017554	051514	000040				
3616	017560	005015	047514	042101	LDRMSG:	.ASCIZ	<15><12>'LOADERS RESTORED'
3617	017566	051105	020123	042522			
3618	017574	052123	051117	042105			
3619	017602	000					
3620	017603	015	041012	042101	PSMSG:	.ASCIZ	<15><12>'BAD PARITY SCAN COMPLETE'
3621	017610	050040	051101	052111			
3622	017616	020131	041523	047101			

3623	017624	041440	040517	046120			
3624	017632	052105	000105				
3625	017636	005015	047514	042101	MLDRSV:	.ASCIZ	<15><12>'LOADERS SAVED IN BANK 1'
3626	017644	051105	020123	040523			
3627	017652	042526	020104	047111			
3628	017660	041040	047101	020113			
3629	017666	000061					
3630	017670	020040	020040	020040	MPCRDR:	.ASCIZ	' - CORE PARITY REGISTER'<15><12>
3631	017676	020040	020055	047503			
3632	017704	042522	050040	051101			
3633	017712	052111	020131	042522			
3634	017720	044507	052123	051105			
3635	017726	005015	000				
3636	017731	040	020040	020040	MPRMGS:	.ASCIZ	' - MOS PARITY REGISTER'<15><12>
3637	017736	020040	026440	046440			
3638	017744	051517	050040	051101			
3639	017752	052111	020131	042522			
3640	017760	044507	052123	051105			
3641	017766	005015	000				
3642						.EVEN	
3643		000001				.END	

CCMFAF0, MEMORY PARITY TEST
CCMFAF.P11 13-JAN-78 12:13

MACY11 30A(1052) 13-JAN-78 12:27 PAGE 79
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0077

EMTDEF	388#	3373	3376	3378	3380	3382	3384	3386	3388	3390								
TESTX	390#	832	884	963	997	1089	1143	1272	1307	1401	1467	1570	1661	1777	1897			
	2083	2303	2415															

. ABS. 017772 000

ERRORS DETECTED: 0

CCMFAF,CCMFAF.LST/CRF/SOL=CCMFAF.P11
RUN-TIME: 4 9 1 SECONDS
RUN-TIME RATIO: 247/15=16.3
CORE USED: 9K (17 PAGES)