

EOF1CCK88880411
CCMFAF.P11

13-JAN-78 12:13

00010000 MEMORY SECURITY TEST MACY (P30A1052) 13-JAN-78 08:07 SE PAGE 1

00010000

780223
SEQ 0001

.REM *

IDENTIFICATION

PRODUCT CODE: AC-7883F-MC
 PRODUCT NAME: CCMFAFD 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 TYPEOUT ROUTINE)

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

5.2.6 TRAPCATCHER

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

EACH VECTOR ENTRANCE ADDRESS IS LOADED WITH THE ADDRESS OF THE NEXT LOCATION. THE NEXT LOCATION IS LOADED WITH A HALT (00000). THUS AN ILLEGAL TRAP OR INTERRUPT WILL CAUSE A HALT AT THE TRAP LOCATION PLUS TWO.

IF A HALT OCCURS IN THE TRAP OR INTERRUPT AREA, EXAMINE REGISTER SIX. IT WILL CONTAIN THE CURRENT STACK ADDRESS. THE CONTENTS OF THE CURRENT STACK ADDRESS IS THE VALUE OF THE LOCATION COUNTER WHEN THE TRAP

OR INTERRUPT OCCURRED.

5.3 PROGRAM AND/OR OPERATOR ACTION

5.3.1 ALTERING THE PARITY MEMORY MAP

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

5.3.2 STOPPING THE PROGRAM

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

6.0 ERRORS

6.1 ERROR PRINTOUTS

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

PC=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 1, KERNEL PAGE ADDRESS REGISTER 1 (ADDRESS 772342)

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

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

6.3 ERROR RECOVERY

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

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

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

7.0 RESTRICTIONS

THE PROGRAM REQUIRES A MINIMUM OF BK MEMORY TO RUN. XXDP CHAINING IS POSSIBLE FOR SYSTEMS GREATER THAN BK.

7.1 STARTING PROCEDURE

PROGRAM MUST BE LOADED INTO LOWER 4K OF MEMORY.

7.2 OPERATING RESTRICTION- AVOID USING THE "HALT" SWITCH

IF THE PROGRAM IS HALTED AT A RANDOM POINT DURING EXECUTION, SEVERAL PROBLEMS MAY ARISE. THE PROGRAM MAY BE RELOCATED TO BANK 1 AT THE TIME IT IS STOPPED, IN WHICH CASE NONE OF THE STANDARD STARTING ADDRESSES WILL WORK. WRITE WRONG PARITY MAY BE SET, IN WHICH CASE

YOU MAY ENTER BAD PARITY WHILE PATCHING. AND MEMORY MAY CONTAIN BAD PARITY SINCE YOU MAY BE IN THE MIDDLE OF A TEST WHICH UTILIZES WRITE WRONG PARITY. IT IS THEREFORE STRONGLY RECOMMENDED THAT YOU HALT THE PROGRAM VIA THE "HALT AT END OF PASS" SWITCH (SW8) OR THE "HALT ON ERROR" SWITCH (SW15) RATHER THAN VIA THE HALT/ENABLE SWITCH.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

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

8.2 STACK POINTERS

THE KERNEL STACK POINTER IS INITIALIZED TO 510.

9.0 PROGRAM DESCRIPTION

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

*

:MEMORY PARITY TEST
:MAINDEC-1'-DCMFA-D
:COPYRIGHT 1973, 1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS.
:AUTHOR: JIM KAPADIA

325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376

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

:SYMBOL DEFINITIONS

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

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

;BIT DEFINITIONS

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

377 000003
 378 000004
 379 000005
 380 000006
 381 000114
 382 177572
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396 000030
 397 000030 016032
 398 000032 000340
 399 000046 000046
 400 000046 011702
 401 000052 000052
 402 000052 040000
 403
 404
 405 000174
 406 000174 000000
 407 000176 000000
 408 000200 000167 001232
 409 000210 000210
 410 000210 000167 014204
 411 000220 000220
 412 000220 000167 001132
 413 000230 000230
 414 000230 000167 001042
 415
 416 000510
 417
 418
 419
 420
 421
 422 000510 000000
 423 000512 000000
 424 000514 000000
 425 000516 000000
 426 000520 000000
 427 000522 000000
 428 000524 000000
 429 000526 000000
 430 000530 000000
 431 000532 000000
 432

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

PARVEC=114
 SRO=177572

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

:MACRO DEFINITIONS

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

;LOAD EMT VECTOR

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

;LOAD STARTING ADDRESS AREA

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

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

:GENERAL DATA AREA

TSTX: 0
 FTITLE: 0
 TEMPX: 0
 ADRPT: 0
 BITPT: 0
 TRFLG: 0
 TYFLG: 0
 TYCOR: 0
 HIADR: 0
 TSTLOC: 0

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

433	000534	000000	SHDBE: 0	: VALUE EXPECTED
434	000536	000000	WAS: 00	: ACTUAL VALUE FOUND
435	000540	000000	TRDATA: 00	
436	000542	000000	MPROK: 00	
437	000544	000000	TBANK: 00	
438	000546	000000	MEMUT: 00	
439	000550	000000	NOKT: 00	: SET TO INDICATE NO KTI1 PRESENT
440	000552	000000	HIWORD: 00	
441	000554	000000	LOWFLG: 00	: IF SET INDICATES TESTING HIGH BYTE
442	000556	000000	ODDFLG: 00	: OF MEMORY LOCATION
443				
444	000560	000000	TEMP: 0	: SET TO INDICATE MAP OF PARITY MEMORY
445	000562	000000	MTYFG: 0	: ALREADY TYPED
446				
447	000564	000000	RELOC: 0	
448				

450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477

```

: MEMORY PARITY CONTROL REGISTER ADDRESSES
: THE LEAST SIGNIFICANT BIT IN THE DEVICE ADDRESS IS SET TO A ONE(1)
: IF THE CONTROL IS FOUND NOT TO BE PRESENT. THE MEMORY PRESENT UNDER
: CONTROL OF EACH CONTROLLER IS REPRESENTED BY 2 OCTAL WORDS. EACH BIT
: REPRESENTS A 4K BLOCK. I.E. BIT0= 0-4K, BIT1= 4-8K, BIT15= 60-64K.
: THE LOW BYTE OF THE LAST WORD FOR EACH REGISTER INDICATES THE OFFSET (0,2,4,OR 6)
: FOR THE FIRST ADDRESS THAT ACTUALLY CORRESPONDED TO THE REGISTER. THE HIGH BYTE GETS
: SET TO 1 TO INDICATE THAT A MEMORY ADDRESS HAS BEEN FOUND FOR THAT REGISTER.
: FOR EXAMPLE, SAY THAT MPRO AND MPR1 EXIST, CONTROLLING INTERLEAVED MEMORY
: FROM 0 TO 16K, AND THAT MPRO CONTROLS THE ADDRESSES ENDING IN 0 AND 4.
: THE MAP WOULD THEN LOOK AS FOLLOWS:
    MPRO: 172100      : BIT 0 IS CLEAR SINCE REGISTER IS PRESENT
           17        : REGISTER CONTROLS 1ST 16K (=4 BANKS)
           0
           400      : LOW BYTE SHOWS THAT FIRST ADDRESS
                    : ENDS IN 0 (OCTAL)
                    : HIGH BYTE CONTAINS A 1 TO INDICATE
                    : THAT AN ADDRESS WAS FOUND
    MPR1: 172102      : BIT 0 IS CLEAR SINCE REGISTER IS PRESENT
           17        : REGISTER CONTROLS 1ST 16K
           0
           402      : LOW BYTE INDICATES THAT THE FIRST
                    : MEMORY ADDRESS ENDS IN 2 (OCTAL)
                    : HIGH BYTE CONTAINS A 1 TO INDICATE
                    : THAT AN ADDRESS WAS FOUND
: THE REST OF THE MAP WOULD APPEAR AS IN THE LISTING
    
```

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	: 0-64K PARITY MEM UNDER THIS CONTROL
483	000600	000000		0	: 64-124K PARITY MEM UNDER THIS CONTROL
484	000602	000000		0	: ADDRESS RESPONSE THIS CONTROL (0,2,4,6)
485	000604	000000		0	
486	000606	172105	MPR2:	172104+1	: 0-64K PARITY MEM UNDER THIS CONTROL
487	000610	000000		0	: 64-124K PARITY MEM UNDER THIS CONTROL
488	000612	000000		0	

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

545
546
547
548
549
550
551
552
553
554
555
556 000770 000000
557 000772 000000
558 000774 000000
559 000776 000000
560 001000 000000
561 001002 000000
562 001004 000000
563 001006 000000
564 001010 000000
565 001012 000000
566 001014 000000
567 001016 000000
568 001020 000000
569 001022 000000
570 001024 000000
571 001026 000000
572 001030 000000
573
574
575 001032 070032
576 001034 077772
577
578
579
580
581
582 001036 125325
583 001040 152652
584 001042 052452
585 001044 025125
586 001046 102070
587 001050 072527
588 001052 177777
589 001054 107030
590 001056 152525
591 001060 000000
592 001062 000000
593
594
595
596 001064 000000
597 001066 000000
598
599
600 001070 000000

: INDICATORS FOR CORE OR MOS PARITY REGISTER:
: EACH INDICATOR REFERS TO A PARTICULAR PARITY REGISTER. IF IT IS
: A CORE PARITY REGISTER THEN A '1' IS STORED IN THE INDICATOR.
: IF IT IS A MOS PARITY REGISTER THEN '-1' GETS STORED.
: EX= IF MPRO (172100) IS FOR CORE AND MPR1 (172102) IS FOR MOS
: THEN THE INDICATOR MAP WILL LOOK AS FOLLOWING:
: INDC0: 000001
: INDC1: 177777

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

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

RESVC: 70032
RESVM: 77772

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

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

: THIS IS A MAP OF THE TOTAL MEMORY PRESENT IN THE SYSTEM.
MEML: 0 ; 0-64K MEM PRESENT IN 4K CONTIGUOUS BLOCKS
MEMH: 0 ; 64-124 MEM PRESENT IN 4K CONTIGUOUS BLOCKS

: THIS IS A MAP OF THE TOTAL PARITY MEMORY PRESENT IN THE SYSTEM.
PMEML: 0 ; 0-64K PARITY MEMORY PRESENT

```

601
602 001072 000000
603
604 001074 000000
605
606
607
608
609
610
611
612
613
614          001100
615 001100 177570
616 001102 177570
617 001104 177564
618 001106 177566
619 001110      000
620 001111      002
621 001112      000
622 001113      000
623
624 001114 105767 177772
625 001120 001401
626 001122 000000
627 001124 010046
628 001126 017600 000002
629 001132 112346
630 001134 001005
631 001136 005726
632 001140 012600
633 001142 062716 000002
634 001146 000002
635 001150 004767 000026
636 001154 122726 000012
637
638 001162 001364
639 001162 016746 177722
640
641 001166 105366 000001
642 001172 002770
643 001174 004767 000002
644 001200 000772
645 001202 105777 177676
646 001206 100375
647 001210 116677 000002 177670
648
649 001216 000207
650
651
652
653
654
655 001220 000000
656
    
```

```

PMEMH: 0
PMEMX: 0
    
```

```

;(IN 4K CONTIGUOUS BLOCKS
;64-124K PARITY MEMORY PRESENT
;(IN 4K CONTIGUOUS BLOCKS)
;TEMP TO HOLD CONTENTS OF EITHER
;LOW OR HIGH MAP
    
```

```

;ROUTINE TO TYPE ASCII MESSAGES. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
    
```

```

          .=1100
SWR:      .WORD   DSWR
DISPLAY:  .WORD   DDISP
TPS:      177564
TPB:      177566
$NULL:    .BYTE   0
$FILLS:   .BYTE   2
$TPFLG:   .BYTE   0
          .BYTE   0
    
```

```

;ADDRESS OF THE SWITCH REGISTER
;ADDRESS OF THE DISPLAY REGISTER
;PRINTER STATUS REGISTER ADDRESS
;PRINTER BUFFER REGISTER ADDRESS
;CONTAINS NULL CHARACTER FOR FILLS
;CONTAINS # OF FILLER CHARACTERS REQUIRED
;"TERMINAL AVAILABLE" FLAG (0=YES)
;RESERVED
    
```

```

$TYPE:    TSTB   $TPFLG
          BEQ    6$
          HALT
6$:        MOV    RO, -(SP)
          MOV    @2(SP), RO
1$:        MOVB  (RO)+, -(SP)
          BNE    2$
          TST   (SP)+
          MOV    (SP)+, RO
7$:        ADD   #2, (SP)
          RTI
2$:        JSR   PC, 5$
3$:        CMPB  #12, (SP)+
          BNE    1$
          MOV   $NULL, -(SP)
4$:        DECB  1(SP)
          BLT   3$
          JSR   PC, 5$
          BR    4$
5$:        TSTB  @TPS
          BPL   5$
          MOVB  2(SP), @TPB
          RTS   PC
    
```

```

;IS THERE A TERMINAL?
;BR IF YES
;HALT HERE IF NO TERMINAL
;SAVE RO
;GET ADDRESS OF ASCII STRING
;PUSH CHARACTER TO BE TYPED ONTO STACK
;BR IF IT ISN'T THE TERMINATOR
;IF TERMINATOR POP IT OFF THE STACK
;RESTORE RO
;ADJUST RETURN PC
;RETURN
;GO TYPE THIS CHARACTER
;CHECK IF THE CHARACTER TYPED
;WAS A LINE FEED
;GO GET NEXT CHARACTER IF NOT LINE FEED
;GET # OF FILLER CHARACTERS NEEDED
;AND THE NULL CHARACTER
;DOES A NULL NEED TO BE TYPED?
;BR IF NO--GO POP THE NULL OF THE STACK
;GO TYPE A NULL
;LOOP
;WAIT UNTIL PRINTER IS READY
;LOAD CHARACTER TO BE
;TYPED INTO DATA REGISTER
    
```

```

;GENERAL DATA AREA
    
```

```

SCNFLG: 0
    
```

```

;SCNFLG GETS SET IF USING
;SCAN ROUTINE (SA=220)
    
```

657 001222 000000
 658 001224 177746
 659 001226 000000
 660 001230 000000
 661 001232 177600
 662 001234 172200
 663 001236 172300
 664 001240 172300
 665 001242 172302
 666 001244 172304
 667 001246 172316
 668 001250 172340
 669 001252 172342
 670 001254 172344
 671 001256 172356
 672 001260 000000
 673 001262 000000
 674 001264 000000
 675 001266 000000
 676 001270 000000
 677 001272 000000
 678 001274 000000
 679
 680
 681
 682

CACHFL: 0 ;CACHE FLAG
 CACHE: 177746 ;CACHE CSR
 KTSTART: 0
 ADRTYP: 0
 PDRTAB: 177600
 PDREND: 172300
 KPDR0: 172300 ;KERNEL PAGE DESCRIPTOR REGISTER ADDRESSES
 KPDR1: 172302
 KPDR2: 172304
 KPDR7: 172316
 KPAR0: 172340 ;KERNEL PAGE ADDRESS REGISTER ADDRESSES
 KPAR1: 172342
 KPAR2: 172344
 KPAR7: 172356
 SPSAV: 0 ;REGISTER SAVE LOCATIONS
 ROSAV: 0
 RISAV: 0
 R2SAV: 0
 R3SAV: 0
 R4SAV: 0
 R5SAV: 0

683 001276 012706 000510
 684 001302 012767 000001 177252
 685 001310 005067 010436
 686 001314 012737 015732 000024
 687 001322 012737 000340 000026
 688 001330 005067 177154
 689 001334 005037 177776
 690 001340 012737 000006 000004
 691 001346 005037 000006
 692 001352 000167 000530
 693
 694
 695
 696
 697

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

698
 699 001356 012706 000510
 700 001362 005767 177124
 701
 702
 703 001366 001006
 704 001370 005267 177624
 705 001374 000167 000232
 706 001400 005067 177614
 707 001404 004767 010346
 708 001410 004767 013100
 709 001414 104000
 710 001416 017603
 711 001420 005767 177124
 712 001424 001002

;ROUTINE TO SCAN ALL MEMORY FOR BAD PARITY AND TYPE 18 BIT ADDRESSES OF BAD
 ;LOCATIONS
 SCAN: MOV #STKPT,SP ;SETUP STACK POINTER
 TST FTITLE ;IF TITLE HAS BEEN PRINTED, REGISTERS
 ;HAVE ALREADY BEEN LOCATED- GO
 ;AND LOCATE IF NOT ALREADY DONE
 BNE SCANB ;BRANCH, REGISTERS HAVE ALREADY BEEN LOCATED
 INC SCNFLG ;INCREMENT SCNFLG
 JMP START1 ;GO TO LOCATE THE REGISTERS
 SCANA: CLR SCNFLG ;RETURN HERE AFTER LOCATING THE REGISTERS
 SCANB: JSR PC,MAPMEM ;SETUP MEMORY MAP
 JSR PC,PSCAN ;SCAN FOR BAD PARITY
 TYPE ;TYPE MESSAGE "BAD PARITY SCAN COMPLETE"
 PSMMSG
 TST NOKT
 BNE .+6

```

713 001426 005037 177572
714 001432 000000
715 001434 000750
716
717
718
719
720
721 001436 012706 000510
722 001442 005067 177114
723 001446 005067 010300
724 001452 012737 015732 000024
725 001460 012737 000340 000026
726 001466 013746 000004
727 001472 013746 000006
728 001476 012737 001512 000004
729 001504 005777 177370
730 001510 000407
731 001512 012767 000176 177360 2$:
732 001520 012767 000174 177354
733 001526 022626
734 001530 012737 001562 000004 4$:
735 001536 012737 000340 000006
736 001544 052737 000014 177746
737 001552 052767 000200 177442
738 001560 000401
739 001562 022626 6$:
740 001564 012637 000006 5$:
741 001570 012637 000004
742 001574 005067 176710
743 001600 005767 176706 1$:
744 001604 001012
745 001606 005267 176700
746 001612 023737 000042 000046
747 001620 001404
748 001622 104000
749 001624 017257
750 001626 104000
751 001630 017636
752
753
754
755
756
757
758
759 001632 104000
760 001634 017026
761 001636 005067 176700
762 001642 012702 000566
763 001646 012703 000770
764 001652 012737 002012 000004
765 001660 005037 000006
766 001664 042712 000001
767 001670 005062 000002
768 001674 005062 000004

```

```

CLR      @#SRO      ;TURN OFF K11 IF PRESENT
HALT
BR       SCAN      ;END OF PARITY SCAN

: NORMAL STARTUP
START:   MOV      #STKPT, SP      ;SET UP STACK POINTER
        CLR      MTYFG          ;CLEAR FLAG WHICH INDICATES MAP TYPED
        CLR      PASCNT         ;INITIALIZE PASS COUNT
        MOV      #PWDRN, @#24    ;SETUP POWER FAIL RETURN
        MOV      #340, @#26
        MOV      @#4, -(SP)      ;SAVE THE ERROR VECTOR
        MOV      @#6, -(SP)
        MOV      @#25, @#4
        TST     @SWR            ;SET UP TIME OUT VECTOR
        BR      4$             ;TRY TO REFERENCE THE SWITCH REGISTER
        ;BRANCH IF NO TIME OUT
        MOV      #SWREG, SWR     ;POINT TO THE SOFTWARE SWITCH REGISTER
        MOV      #DISPREG, DISPLAY ;POINT TO THE SOFTWARE DISPLAY REG.
        CMP     (SP)+, (SP)+     ;RESTORE THE STACK POINTER
        MOV      @#5, @#4
        MOV      #340, @#6
        BIS     #14, @#177746    ;DISABLE CACHE
        BIS     #200, CACHFL     ;SET FLAG FOR CACHE
        BR      5$
        CMP     (SP)+, (SP)+
        MOV     (SP)+, @#6      ;RESTORE ERROR VECTOR
        MOV     (SP)+, @#4
        CLR     TSTX
        TST     FTITLE          ;IS TITLE PRINTED YET?
        BNE    START1          ;YES, SKIP OVER
        INC    FTITLE           ;SET FLAG
        CMP     @#42, @#46      ;ARE WE IN ACT11 AUTOMATIC MODE?
        BEQ   START1          ;YES SKIP TITLE
        TYPE   "MEMORY PARITY TEST" ;TYPE TITLE "MEMORY PARITY TEST"
        MTIT  "MAINDEC-11-DCMFA" ;MAINDEC-11-DCMFA"

        ;TYPE "LOADERS SAVED IN BANK 1.

:SEARCH FOR PARITY REGISTERS PRESENT AND TYPE ADDRESSES OF THOSE FOUND
:FAILURE TO LOCATE A REGISTER INDICATES THAT THE ADDRESS TIMED OUT OR THAT
:BITS 5-7 IN THE REGISTER DID NOT SET
START1:  TYPE   "MEMORY PARITY REGISTERS PRESENT ARE:"
        MPRB
        CLR     MPROK           ;CLEAR MPR FLAG
        MOV     #MPRO, R2       ;SET UP POINTERS
        MOV     #INDCO, R3      ;POINTER TO CORE-MOS
        MOV     #GMPRB, @#4     ;SET UP TIMEOUT TRAP RETURN
        CLR     @#6
        BIC    @#1(2)          ;CLEAR FLAG BIT IN TABLE
        CLR    2(R2)           ;INITIALIZE LOCATIONS IN THE TABLE
        CLR    4(R2)

```

```

763 001700 005062 000006
770 001704 005772 000000
771 001710 052772 000340 000000
772 001716 032772 000340 000000
773 001724 001414
774 001726 011267 176562
775 001732 004567 013460
776 001736 000514
777 001740 017670
778 001742 000006
779 001744 104000
780 001746 017670
781 001750 012713 000001
782 001754 000413
783 001756 011267 176532
784 001762 004567 013430
785 001766 000514
786 001770 017731
787 001772 000006
788 001774 104000
789 001776 017731
790 002000 012713 177777
791 002004 005267 176532
792 002010 000403
793 002012 022626
794 002014 052712 000001
795 002020 062702 000010
796 002024 005723
797 002026 020227 000766
798 002032 002714
799 002034 012737 000006 000004
800 002042 005767 177152
801
802 002046 001402
803 002050 000167 177324
804
805 002054 005767 176462
806 002060 001012
807 002062 104000
808 002064 017212
809 002066 005737 000042
810 002072 001402
811 002074 000167 007572
812 002100 000000
813 002102 000167 177330
814
815 002106 012767 002156 014346
816 002114 105767 177102
817 002120 001403
818 002122 012777 000014 177074
819 002130 004767 012146
820 002134 012767 000100 014314
821 002142 012737 000006 000004
822 002150 005067 176344
823
824

```

```

CLR 6(R2)
TST 2(2)
BIS #340,2(2)
BIT #340,2(2)
BEQ 1$
MOV (2),TEMPX
JSR RS,CACNV
TEMPX
MPRCOR
6
TYPE
MPRCOR
MOV #1,(R3)
BR 2$
MOV (2),TEMPX
JSR RS,OACNV
TEMPX
MPRMOS
6
TYPE
MPRMOS
MOV #-1,(R3)
INC MPROK
BR GMPRC
GMPRB: CMP (SP)+,(SP)+
BIS #1,R2
GMPRC: ADD #10,R2
TST (R3)+
CMP R2,#TREG
BLT GMPRA
MOV #6,2#4
TST SCNFLG
BEQ GMPRD
JMP SCANA
GMPRD: TST MPROK
BNE BEGIN
NOREG: TYPE
MTR
TST 2#4
BEQ .+6
JMP LOGICAL
HALT
JMP START
BEGIN: MOV #TEST1+2,RETURN
TSTB CACHFL
BEQ .+10
MOV #14,2CACHE
JSR PC,SAVLD
MOV #100,IMAX
MOV #6,2#4
CLR BITPT

```

```

: DOES THIS MPR EXIST? (IF NO, TIMES OUT)
: YES- IS IT AN AF11-LP OR MA11-P CORE PARITY REG
: NO, IS IT A MOS-11 PARITY REGISTER? BRANCH
: YES- PRINT REGISTER ADDRESS
: (GET ASCII)
: (TYPE ADDRESS)
: SET INDICATOR FOR CORE PARITY
: IT IS A MOS REGISTER, PRINT ADDRESS
: (GET ASCII)
: (TYPE ADDRESS)
: SET INDICATOR FOR MOS PARITY
: SET MPR REGISTER PRESENT FLAG
: SKIP NEXT
: RESTORE STACK POINTER
: SET FLAG INDICATING REGISTER NOT PRESENT
: UPDATE POINTER
: DONE YET?
: NO LOOP
: YES, RESTORE TRAPCATCHER
: ARE YOU IN THE ROUTINE TO SCAN
: MEMORY FOR BAD PARITY-(SCAN)
: NO BRANCH TO CARRY ON NORMALLY
: YES, GO BACK TO THE MEMORY
: SCAN ROUTINE
: ANY PARITY REGISTERS PRESENT?
: YES- GO TEST CONTROLS PRESENT
: NO- TYPE "NO PARITY REGISTER FOUND"
: LOADED BY MONITOR?
: NO, BRANCH
: YES- EXIT, NO REGISTERS PRESENT
: NO REGISTERS TO TEST
: IF CONTINUED, TRY AGAIN
: SETUP SCOPE RETURN
: CACHE ?
: BRANCH IF NO
: DISABLE CACHE
: SAVE MONITOR
: MAXIMUM ITERATION COUNT
: RESTORE TRAPCATCHER IN TIMEOUT VECTOR

```

```

825
826
827
828
829
830
831 002154 104001
832 002156 012777 000001 176716
833 002164 012700 000566
834 002170 012704 000770
835 002174 032710 000001 1$:
836 002200 104042
837 002202 C.1001
838
839 002204 022714 000001
840 002210 001004
841 002212 016767 176614 176610
842 002220 000403
843 002222 016767 176606 176600 5$:
844 002230 012702 000U01
845
846 002234 005011
847 002236 011167 176524
848 002242 046767 176562 176516
849 002250 001401
850 002252 104002
851
852
853
854 002254 030267 176550 2$:
855 002260 001010
856
857 002262 010211
858 002264 011103
859 002266 005011
860 002270 046703 176534
861
862 002274 020203
863 002276 001401
864 002300 104002
865
866
867
868 002302 006302 3$:
869 002304 103363
870
871 002306 062700 000010 4$:
872 002312 005724
873
874 002314 020027 000766
875 002320 002725
876 002322 005067 176440
877
878
879
880

```

```

;*****
;SHOW THAT BITS 0,2,5-11, AND 15 OF EACH CORE PARITY REGISTER PRESENT
;CAN BE SET AND CLEARED. BITS 0,2,15 OF EACH MOS PARITY REGISTER
;PRESENT CAN BE SET AND CLEARED
;*****
TEST1: SCOPE
MOV #1, @DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
MOV #MPRO, R0 ;LOAD ADDRESS OF TABLE INTO R0
MOV #INDCO, R4 ;LOAD ADDRESS OF INDICATOR IN R4
BIT #1, @R0 ;IS THIS REGISTER PRESENT?
BNE 4$, ;NO- BRANCH TO GET NEXT ADDRESS
MOV @R0, R1 ;YES- LOAD R1 WITH ADDRESS OF
;PARITY REGISTER
CMP #1, (R4) ;IS THIS REGISTER CORE?
BNE 5$, ;NO
MOV RESVC, RESRVD ;YES, CORE. STORE RESERVED BITS
BR .+10
MOV RESVM, RESRVD ;MOS. STORE RESERVED BITS
MOV #1, R2 ;LOAD R2 WITH VALUE OF FIRST BIT
;TO BE TESTED
CLR @R1 ;INITIALIZE PARITY REGISTER
MOV @R1, TREG ;READ CONTENTS OF PARITY REGISTER
BIC RESRVD, TREG ;CLEAR BITS WHICH ARE RESERVED
BEQ .+4 ;CHECK OTHER BITS- BRANCH IF OK
ERROR ;CLEAR INSTRUCTION DID NOT INITIALIZE
;ALL USED BITS IN PARITY REGISTER
;TO ZERO (R1 CONTAINS ADDRESS OF
;FAILING REGISTER)
BIT R2, RESRVD ;IS THIS BIT RESERVED?
BNE 3$, ;YES- DON'T TEST IT SINCE IT
;MAY BE ZERO OR ONE
MOV R2, @R1 ;NO- SET THIS BIT IN THE PARITY REGISTER
MOV @R1, R3 ;READ AND SAVE CONTENTS OF PARITY REGISTER
CLR @R1 ;CLEAR PARITY REGISTER
BIC RESRVD, R3 ;CLEAR BIT LOCATIONS THAT ARE
;RESERVED
CMP R2, R3 ;CHECK REST
BEQ .+4 ;BRANCH IF OK
ERROR ;PARITY REGISTER WHOSE ADDRESS IS IN R1
;WAS INCORRECT AFTER THE VALUE IN R2
;WAS WRITTEN INTO IT. ACTUAL CONTENTS
;(WITH UNUSED BITS CLEARED) IS IN R3
ASL R2 ;ROTATE BIT TO BE TESTED
BCC 2$, ;IF NOT DONE WITH ALL BIT POSITIONS
;GO TEST THIS ONE
ADD #10, R0 ;MOVE R0 TO POINT TO NEXT POSSIBLE ADDRESS
TST (R4)+ ;
;OF A PARITY REGISTER
;AT END OF TABLE?
CMP R0, #TREG ;NO, BRANCH
BLT 1$,
CLR TREG
;*****
;SHOW THAT RESET CLEARS BITS 0,2, AND 15 OF EACH PARITY REGISTER

```



```

881 ;PRESENT.
882 ;*****
883 TEST2: SCOPE
884 002326 104001 000002 176544 MOV #2, @DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
885 002330 012777 014114 CLR IMAX ;DON'T ITERATE TEST
886 002336 005067 000566 MOV #MPRO, R0 ;LOAD POINTER
887 002342 012700 000770 MOV #INCC, R3 ;POINTER TO INDICATOR
888 002346 012703 000770 BIT #1, @R0 ;IS THIS PARITY REGISTER PRESENT?
889 002352 032710 000001 15: BNE 5$ ;NO BRANCH
890 002356 001012 000001 CMP #1, (R3) ;IS THIS CORE OR MOS PARITY REGISTER?
891 002360 022713 BEQ 6$ ;NO BRANCH
892 002364 001404 MOV #100015, @R0 ;MOS-SET ALL DEFINED BITS TO 1
893 002366 012770 100015 000000 BR .+10
894 002374 000403 6$: MOV #107745, @R0 ;CORE- SET ALL DEFINED BITS TO 1
895 002376 012770 107745 000000 5$: ADD #10, R0 ;MOVE POINTER TO POINT TO NEXT MPR ADDRESS
896 002404 062700 000010 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 ;NO- CONTINUE
901 002420 105767 176466 TSTB $TPFLG ;YES- TERMINAL AVAILABLE?
902 002424 001003 4$ 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$: RESET ;ISSUE INIT
906 002436 012700 000566 MOV #MPRO, R0 ;LOAD ADDRESS OF THE TABLE
907 002442 012703 000770 MOV #INCC, R3 ;POINTER TO INDICATOR
908 002446 105767 176550 TSTB CACHFL ;CACHE ?
909 002452 001403 BEQ 2$ ;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 7$ 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 WMP 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 002520 000411 BR 3$-4
927 002522 017002 000000 7$: MOV @R0, R2 ;MOS GET CONTENTS OF REGISTER
928 002526 005070 000000 CLR @R0 ;MAKE SURE THAT WMP AND AE ARE CLEAR
929 002532 046702 176276 BIC RESVM, R2 ;MASK RESERVED BITS FOR MOS PAR REG
930 002536 005702 TST R2 ;CHECK REST
931 002540 001401 BEQ .+4 ;RESET DID CLEAR ALL BITS
932 002542 104002 ERROR ;MOS PARITY REGISTER WHOSE ADDRESS IS
933 ;POINTED BY R0 WAS INCORRECT. AFTER
934 ;ISSUING RESET CONTENTS OF PAR REG
935 ;WERE AS SHOWN IN R2(UNUSED BITS
936 ;HAVE BEEN MASKED)

```

```

937 002544 005070 000000
938 002550 062700 000010
939
940 002554 005723
941 002556 020027 000766
942 002562 002737
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962 002564 104001
963 002566 012777 000003 176306
964 002574 005767 175762
965 002600 001044
966 002602 004767 011204
967 002606 004767 007144
968 002612 004767 007466
969
970
971 002616 005067 176246
972 002622 005067 176244
973 002626 012701 000566
974 002632 032711 000001
975 002636 001006
976 002640 056167 000002 176222
977 002646 056167 000004 176216
978 002654 062701 000010
979 002660 020127 000766
980 002664 103762
981 002666 004767 010126
982 002672 005267 175664
983 002676 032777 001000 176174
984 002704 001402
985 002706 000000
986
987 002710 000742
988
989
990
991
992

```

```

3$: CLR 2(R0) ;REINITIALIZE PARITY REGISTER
ADD #10,R0 ;MOVE POINTER TO POINT TO ADDRESS
;OF NEXT REGISTER
TST (R3)+ ;INCREMENT POINTER TO INDICATOR
CMP R0,#TREG ;DONE?
BLT 2$ ;NO- LOOP

```

```

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

```

```

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

```

```

*****
SHOW THAT ASSERT PB WORKS CORRECTLY FOR EACH REGISTER
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 TEST4: SCOPE                                ;LOAD THE TEST NUMBER INTO THE DISPLAY
997     MOV     #4,@DISPLAY
998     MOV     #100,IMAX
999     JSR     %7,CLAPAR                        ;CLEAR ALL PARITY REGISTERS
1000    TST     NOKT
1001    BNE     IS
1002    JSR     %7,NRALL
1003    JSR     %7,MAP1                          ;KT11 PRESENT?
1004                                         ;NO-BRANCH
1005                                         ;YES-MAP ALL PAGES NON RESIDENT
1006     MOV     #MPRO,RO
1007     MOV     #INCO,R2
1008     BIT     #1,@RO
1009     BEQ     TST4
1010     LOP4:  ADD     #10,RO
1011             TST     (R2)+
1012             BLO     LOOP4
1013             BR     DONE4
1014     TST4:  JSR     %7,LOCATM                ;THEN MAP KERNEL 0 TO BANK 0, KERNEL
1015                                         ;7 TO EXTERNAL BANK, SET KERNEL
1016                                         ;0,1, AND 7 R4, AND TURN ON KT11
1017                                         ;SETUP TO FIND REGISTERS PRESENT
1018                                         ;IS THIS REGISTER PRESENT?
1019                                         ;YES-BRANCH TO TEST IT
1020                                         ;NO-CHECK FOR ANOTHER ONE
1021                                         ;INCREMENT PTR9LT
1022                                         ;BRANCH WHEN ALL REGISTERS HAVE BEEN TESTED
1023                                         ;LOCATE MEMORY CORRESPONDING TO
1024                                         ;THIS REGISTER- IF NO KT11, R1 SHOULD
1025                                         ;BE RETURNED CONTAINING THE ADDRESS
1026                                         ;OF THE 1ST LOCATION CONTROLLED BY THIS
1027                                         ;REGISTER (INCLUDING EXTERNAL INTERLEAVE
1028                                         ;OFFSET IF NEEDED)
1029                                         ;IF KT11 IS PRESENT, R1 SHOULD BE RETURNED
1030                                         ;POINTING TO THE 1ST LOCATION CONTROLLED
1031                                         ;BY THIS REGISTER, MAPPED THRU KERNEL
1032                                         ;PAGE 1. KERNEL PAGE 1 SHOULD BE
1033                                         ;MAPPED TO THE CORRECT BANK
1034                                         ;IS ERROR RETURN INDICATED?
1035                                         ;NO-BRANCH
1036                                         ;MAP INDICATES NO PARITY MEMORY
1037                                         ;IS CONTROLLED BY THIS REGISTER
1038                                         ;RO POINTS TO THE ADDRESS OF THE
1039                                         ;PARITY REGISTER
1040                                         ;SETUP PARITY TRAP RETURN
1041                                         ;SET WRITE WRONG PARITY
1042                                         ;WRITE CONTENTS OF LOCATION WITH
1043                                         ;WRONG PARITY
1044                                         ;CLEAR PARITY REGISTER
1045                                         ;READ BAD PARITY WITH ACTION ENABLE
1046                                         ;CLEARED- NO TRAP EXPECTED
1047                                         ;CHANGE PARITY TRAP RETURN
1048                                         ;SET ACTION ENABLE WITH PARITY ERROR
1049                                         ;ALREADY SET- SHOULDN'T TRAP YET
1050                                         ;CHANGE PARITY TRAP RETURN
1051                                         ;READ LOCATION AGAIN- SHOULD GET
1052                                         ;A PARITY TRAP DUE TO READING BAD
1053                                         ;PARITY WITH ACTION ENABLE SET
1054                                         ;NO PARITY TRAP AFTER READING LOCATION
1055                                         ;WHICH SHOULD CONTAIN BAD PARITY-
1056                                         ;R1 CONTAINS ADDRESS OF MEMORY LOCATION

```

```

1049
1050
1051
1052 003074 005070 000000 CONT4: CLR 2(RO)
1053 003100 005511 ADC 2R1
1054 003102 005070 000000 CLR 2(RO)
1055 003106 000730 BR LOP4
1056 003110 104002 TRP4A: ERROR
1057
1058
1059
1060
1061 003112 022626 CMP (SP)+,(SP)+
1062 003114 000767 BR CONT4
1063 003116 104002 TRP4B: ERROR
1064
1065
1066
1067
1068
1069
1070 003120 022626 CMP (SP)+,(SP)+
1071 003122 000764 BR CONT4
1072 003124 005770 000000 TRP4C: TST 2(RO)
1073 003130 100401 BMI .+4
1074 003132 104002 ERROR
1075
1076
1077 003134 022626 CMP (SP)+,(SP)+
1078 003136 000756 BR CONT4
1079 003140 012737 000116 000114 DONE4: MOV #PARVEC+2,2#PARVEC ;RESTORE TRAP CATCHER
1080 003146 005767 175376 TST NOKT
1081 003152 001002 BNE .+6
1082 003154 005037 177572 CLR 2#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 MOV #5,2DISPLAY ;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,RO ;SETUP TO FIND REGISTERS PRESENT
1098 003216 032710 000001 LOPS: BIT #1,2RO ;IS THIS REGISTER PRESENT?
1099 003222 001406 BEQ T$5 ;YES- BRANCH TO TEST IT
1100 003224 062700 000010 LOPS: ADD #10,RO ;NO- CHECK FOR ANOTHER ONE
1101 003230 020027 000766 CMP RO,2TREG
1102 003234 103770 BLO LOPS
1103 003236 000431 BR DONS
1 003240 004767 010662 T$5: JSR PC,LOCATM ;EXIT WHEN ALL REGISTERS HAVE BEEN TESTED
;LOCATE MEMORY CORRESPONDING TO THIS

```

```

1105
1106
1107
1108
1109 003244 032701 000001
1110 003250 001403
1111 003252 104002
1112
1113
1114
1115 003254 000167 177744
1116 003260 012770 000004 000000
1117 003266 011111
1118
1119 003270 005711
1120 003272 042770 000004 000000
1121 003300 011111
1122 003302 005711
1123 003304 005770 000000
1124 003310 100401
1125 003312 104002
1126
1127
1128
1129 003314 005070 000000
1130 003320 000741
1131 003322 005767 175222
1132 003326 001002
1133 003330 005037 177572
1134
1135
1136
1137
1138
1139
1140
1141
1142 003334 104001
1143 003336 012777 000006 175536
1144 003344 004767 010442
1145 003350 005767 175174
1146 003354 001004
1147 003356 004767 010330
1148 003362 004767 010474
1149
1150 003366 012700 000566
1151 003372 032710 000001
1152 003376 001406
1153 003400 062700 000010
1154 003404 020027 000766
1155 003410 103770
1156 003412 000523
1157
1158 003414 004767 010506
1159
1160

```

```

BIT #1,R1
BEQ .+10
ERROR

JMP LOP5
MOV #WMP,2(RO)
MOV #R1,#R1

TST #R1
BIC #WMP,2(RO)
MOV #R1,#R1
TST #R1
TST 2(RO)
BMI .+4
ERROR

CLR 2(RO)
BR LOP5
TST NOKT
BNE .+6
CLR #SRO

```

```

;REGISTER- R1 WILL BE RETURNED CONTAINING
;THE ADDRESS OF THE FIRST LOCATION
;CONTROLLED BY THIS REGISTER (MAPPED
;THRU KERNEL PAGE 1 IF KT11 IS PRESENT)
;IS ERROR RETURN INDICATED?
;NO- BRANCH
;MAP INDICATES NO PARITY MEMORY IS
;CONTROLLED BY THIS REGISTER. RO
;POINTS TO THE ADDRESS OF THE
;PARITY REGISTER

;SET WMP IN THIS REGISTER
;WRITE CONTENTS OF LOCATION WITH
;WRONG PARITY
;DETECT WRONG PARITY
;CLEAR WMP IN PARITY REGISTER
;RESTORE GOOD PARITY
;REREAD LOCATION
;READ CONTENTS OF PARITY REGISTER
;BRANCH IF PARITY ERROR IS STILL SET
;PARITY ERROR BIT CLEARED BY READING
;GOOD PARITY (OR POSSIBLY WHILE DOING
;THE BIC TO CLEAR WMP). RO POINTS TO
;THE PARITY REGISTER ADDRESS.
;CLEAR THE PARITY REGISTER

;TURN OFF KT11 IF PRESENT

```

```

*****
;SHOW THAT PARITY GENERATE AND DETECT LOGIC WORKS CORRECTLY FOR EACH BYTE
;SHOW THAT WRITE WRONG PARITY WORKS FOR HIGH AND LOW BYTES
;SHOW THAT WRITING INTO LOCATION WHEN WRITE WRONG PARITY IS NOT SET
;RESTORES GOOD PARITY
*****

```

```

TEST6: SCOPE
MOV #6,2DISPLAY
JSR %7,CLAPAR
TST NOKT
BNE 1$
JSR %7,NRALL
JSR %7,MAP1

1$: MOV #MPRO,RO
LOOP6: BIT #1,RO
BEQ TEST6
LOP6: ADD #10,RO
CMP RO,#STREG
BLO LOOP6
BR DONE6

TEST6: JSR %7,LOCATM

```

```

;LOAD THE TEST NUMBER INTO THE DISPLAY
;CLEAR ALL PARITY REGISTERS
;KT11 PRESENT?
;NO BRANCH
;YES- MAP IT (KERNEL 0 TO BANK 0, RW;
;KERNEL 7 TO EXTERNAL BANK, RW; KERNEL 1 RW)
;AND TURN IT ON
;SETUP TO FIND REGISTERS PRESENT
;IS THIS REGISTER PRESENT?
;YES- BRANCH TO TEST IT
;NO- CHECK FOR ANOTHER ONE

;BRANCH TO DONE IF ALL REGISTERS
;HAVE BEEN TESTED
;LOCATE MEMORY CORRESPONDING TO
;THIS REGISTER- R1 SHOULD BE RETURNED
;CONTAINING THE ADDRESS OF THE FIRST

```

```

1161
1162
1163 003420 032701 000001
1164 003424 001403
1165 003426 104002
1166
1167
1168
1169
1170 003430 000167 177744
1171
1172
1173 003434 005011
1174 003436 005070 000000
1175 003442 005002
1176
1177 003444 110211
1178 003446 005711
1179 003450 005770 000000
1180 003454 100006
1181 003456 104002
1182
1183
1184
1185 003460 005070 000000
1186 003464 005011
1187 003466 005002
1188 003470 000402
1189 003472 105202
1190 003474 001363
1191 003476 110261 000001
1192 003502 005711
1193 003504 005770 000000
1194 003510 100002
1195 003512 104002
1196
1197
1198 003514 000402
1199 003516 105202
1200 003520 001366
1201
1202
1203
1204 003522 005011
1205 003524 005002
1206 003526 012770 000004 000000
1207 003534 110211
1208 003536 005070 000000
1209
1210 003542 005711
1211 003544 005770 000000
1212 003550 100402
1213 003552 104002
1214
1215
1216

```

```

BIT #1 R1
BEG .+10
ERROR

JMP LOP6

; FIRST SHOW THAT IF THE PARITY REGISTER IS CLEARED INITIALLY,
; PARITY ERROR DOESN'T SET
CLR @R1
CLR @ (R0)
CLR R2

1$: MOVB R2 @R1
TST @R1
TST @ (R0)
BPL 5$
ERROR

CLR @ (R0)
CLR @R1
CLR R2
BR 2$

5$: INCB R2
BNE 1$

2$: MOVB R2 1(R1)
TST @R1
TST @ (R0)
BPL .+6
ERROR

BR 6$
INCB R2
BNE 2$

6$: CLR @R1
CLR R2
MOV @R1 @ (R0)
MOV R2 @R1
CLR @ (R0)

TST @R1
TST @ (R0)
BMI .+6
ERROR

```

```

; LOCATION CONTROLLED BY THIS REGISTER
; (VIRTUAL THRU KERNEL PAGE 1 IF KTI1 PRESENT
; IF NO MEMORY WAS FOUND TO CORRESPOND
; ODD ADDRESS IS RETURNED-BRANCH IF OK
; MAP INDICATES NO PARITY MEMORY IS
; CONTROLLED BY THIS REGISTER
; R0 POINTS TO THE ADDRESS OF THE
; PARITY REGISTER
; AFTER ERROR, CHECK FOR NEXT REGISTER

; INITIALIZE LOCATION UNDER TEST
; INITIALLY CLEAR PARITY REGISTER
; R2 CONTAINS VALUE TO BE LOADED
; INTO MEMORY
; WRITE VALUE INTO LOW BYTE
; READ WORD TO CHECK PARITY
; CHECK PARITY REGISTER
; BRANCH IF ERROR NOT SET
; PARITY ERROR SET WHEN VALUE IN R2 WAS
; WRITTEN AND READ BACK FROM LOW BYTE OF
; LOCATION WHOSE ADDRESS IS CONTAINED IN
; R1 (LWP WAS NOT SET)
; CLEAR ERROR BIT
; REINITIALIZE TEST LOCATION
; REINITIALIZE VALUE TO BE USED

; INCREMENT VALUE TO BE LOADED
; LOOP UNTIL ALL VALUES HAVE BEEN USED
; WRITE ALL VALUES INTO HIGH BYTE
; READ WORD TO CHECK PARITY
; CHECK PARITY REGISTER
; BRANCH IF ERROR NOT SET
; PARITY ERROR SET WHEN VALUE IN R2
; WAS WRITTEN AND READ BACK FROM HIGH BYTE
; OF LOCATION WHOSE ADDRESS IS CONTAINED
; IN R1 (LWP WAS NOT SET)
; INCREMENT VALUE TO BE LOADED
; LOOP UNTIL ALL VALUES HAVE BEEN USED

; TEST PARITY GENERATE AND DETECT LOGIC BY SETTING WRITE WRONG PARITY AND
; WRITING EACH POSSIBLE VALUE TO THE LOW BYTE THEN TO THE HIGH BYTE
6$: CLR @R1
CLR R2
MOV @R1 @ (R0)
MOV R2 @R1
CLR @ (R0)

; INITIALIZE LOCATION UNDER TEST
; INITIALIZE VALUE TO BE WRITTEN
; SET WRITE WRONG PARITY
; WRITE WRONG PARITY IN LOW BYTE
; CLEAR WRITE WRONG PARITY, AND CLEAR
; PARITY ERROR IF SET
; READ BACK WRONG PARITY
; PARITY ERROR SET?
; YES-BRANCH
; PARITY ERROR DID NOT SET WHEN THE
; LOCATION UNDER TEST WAS WRITTEN
; AND READ BACK WITH WRITE WRONG PARITY
; SET. R0 POINTS TO ADDRESS OF PARITY

```

```

1217
1218
1219
1220
1221 003554 000402 BR .+6
1222 003556 105202 INCB R2
1223 003560 001362 BNE 3$
1224 003562 005011 CLR @R1
1225 003564 005070 CLR @ (R0)
1226 003570 005711 TST @R1
1227
1228 003572 005770 000000 TST @ (R0)
1229 003576 100001 BPL .+4
1230 003600 104002 ERROR
1231
1232
1233 003602 012770 000004 000000 4$: MOV #WWP,@ (R0)
1234 003610 110261 000001 MOVB R2,@ (R1)
1235 003614 005070 000000 CLR @ (R0)
1236
1237 003620 005711 TST @R1
1238 003622 005770 000000 TST @ (R0)
1239 003626 100402 BMI .+6
1240 003630 104002 ERROR
1241
1242
1243
1244
1245
1246
1247
1248
1249 003632 000402 BR .+6
1250 003634 105202 INCB R2
1251 003636 001361 BNE 4$
1252 003640 005011 CLR @R1
1253 003642 005070 000000 CLR @ (R0)
1254 003646 005711 TST @R1
1255
1256 003650 005770 000000 TST @ (R0)
1257 003654 100001 BPL .+4
1258 003656 104002 ERROR
1259
1260
1261 003660 000647 BR LOP6
1262 003662 005767 174662 DONE6: TST NOKT
1263 003666 001002 BNE .+6
1264 003670 005037 177572 CLR @#SRO
1265
1266
1267
1268
1269
1270
1271 003674 104001 TEST7: SCOPE
1272 003676 012777 000007 175176 MOV #7,@DISPLAY

```

```

: REGISTER, R1 CONTAINS ADDRESS OF LOCATION
: BEING TESTED (VIRTUAL THRU KERNEL
: PAGE 1 IF KT11 IS PRESENT). R2
: CONTAINS THE VALUE WRITTEN
: EXIT LOOP AFTER ERROR
: INCREMENT DATA
: LOOP TILL DONE WITH ALL VALUES
: REINITIALIZE LOCATION TO CLEAR BAD PARITY
: CLEAR ERROR IF SET
: READ LOCATION, WHICH SHOULD NOW HAVE
: GOOD PARITY
: PARITY ERROR SET?
: NO BRANCH
: GOOD PARITY WAS NOT RESTORED BY
: WRITING INTO THE LOCATION WITH
: WRITE WRONG PARITY CLEARED
: SET WRITE WRONG PARITY
: WRITE WRONG PARITY IN HIGH BYTE
: CLEAR WRITE WRONG PARITY AND PARITY
: ERROR IF SET
: READ BACK WRONG PARITY
: PARITY ERROR SET?
: YES-BRANCH
: PARITY ERROR DID NOT SET WHEN THE LOCATION
: UNDER TEST WAS WRITTEN AND READ BACK WITH
: WRITE WRONG PARITY SET. R0 POINTS
: TO THE ADDRESS OF THE PARITY REGISTER.
: THE VALUE IN R2 WAS WRITTEN INTO THE HIGH
: BYTE OF THE LOCATION WHOSE ADDRESS IS IN R1
: (VIRTUAL THRU KERNEL PAGE 1 IF KT11 PRESENT)
: THEN THE PARITY REGISTER WAS
: CLEARED AND THE WORD WAS READ BACK
: EXIT LOOP AFTER ERROR
: INCREMENT DATA
: LOOP TILL DONE WITH ALL VALUES
: CLEAR BAD PARITY IN TEST LOCATION
: CLEAR PARITY ERROR BIT IF SET
: READ LOCATION, WHICH SHOULD NOW
: HAVE GOOD PARITY
: CHECK PARITY ERROR BIT
: BRANCH IF CLEAR
: WRITING INTO LOCATION WHEN WRITE
: WRONG PARITY WAS NOT SET DID NOT
: WRITE GOOD PARITY
: GO CHECK FOR ANOTHER PARITY REGISTER

; TURN OFF KT11 IF PRESENT

```

```

:*****
: SHOW THAT SETTING PARITY ERROR AFTER SETTING ACTION ENABLE WON'T CAUSE A TRAP
:*****
TEST7: SCOPE ;LOAD THE TEST NUMBER INTO THE DISPLAY
MOV #7,@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,RO      ;SETUP TO GET ADDRESS OF REGISTER PRESENT
1277 003726 032710 000001      LUP7:   BIT      #1,@RO
1278 003732 001406      BEQ      TST7          ;BRANCH TO TEST REGISTER
1279 003734 062700 000010      LOOP7:  ADD      #10,RO
1280 003740 020027 000766      CMP      RO,#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,@(RO) ;SET ACTION ENABLE
1284 003756 052770 100000 000000  BIS      #PEAR,@(RO)    ;SET PARITY ERROR
1285 003764 000240      NOP
1286 003766 005070 000000      CLR      @(RO)        ;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
1290 004004 104002      TRP7:   ERROR        TEST10
1291
1292
1293
1294
1295 004006 022626      CMP      (SP)+,(SP)+  ;TRAP OCCURRED WHEN PARITY ERROR BIT
1296 004010 005070 000000      CLR      @(RO)        ;WAS SET VIA A BIS INSTRUCTION
1297 004014 000747      BR      LOOP7        ;WITH ACTION ENABLE ALREADY SET.
1298
1299
1300
1301
1302
1303
1304
1305

```

```

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

```



```

1362                                     ; REFERENCED TO CAUSE A PARITY TRAP
1363                                     ; (VIRTUAL IF KT11 IS PRESENT)
1364 004230 012716 004164          2$:  MOV      #INST1, @SP          ; GO EXECUTE INSTRUCTION 1 AGAIN
1365 004234 000002                                     ;
1366 004236 012737 004252 000114  1$:  MOV      #TRP10A, @#PARVEC      ; CHANGE PARITY TRAP RETURN
1367 004244 012716 004172                                     ; GO EXECUTE INSTRUCTION 2
1368 004250 000002                                     ;
1369 004252 022712 000001          TRP10A: CMP      #1, (R2)          ; IS THIS A CORE REG?
1370 004256 001005                                     ; NO, BRANCH
1371 0C4260 032770 000040 000000  BIT      #BITS, @ (R0)      ; PARITY TRAP OCCURRED- CHECK PARITY
1372                                     ; ERROR ADDRESS BITS
1373 004266 001001                                     ; 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                                     ; R0 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 0C4272 022626                                     ;
1381 004274 012737 000116 000114  1$:  CONT10: CMP      (SP)+, (SP)+      ; RESTORE TRAPCATCHER
1382 004302 005070 000000          MOV      #PARVEC+2, @#PARVEC      ; CLEAR PARITY REGISTER
1383 004306 005511          CLR      @ (R0)          ; CLEAR BAD PARITY
1384 004310 005561 004000          ADC      @R1
1385 004314 005070 000000          ADC      4000 (R1)
1386 004320 000662          CLR      @ (R0)          ; CLEAR PARITY ERROR BIT IF SET
1387 004322 005767 174222          DONE10: BR      LOOP10
1388 004326 001002          TST      NOKT
1389 004330 005037 177572          BNE     +6
1390 004334 000401          CLR     @#SR0          ; TURN OFF KT11 IF PRESENT
1391 004336 000000          BR      TEST11
1392          COUNT: 0
1393
1394
1395          ; *****
1396          ; IF MULTIPLE PARITY ERRORS OCCUR DURING ONE INSTRUCTION (WITH ACTION ENABLE
1397          ; NOT SET) THE ERROR ADDRESS BITS WILL RECORD THE LAST ERROR (ONLY FOR CORE PARITY
1398          ; REGISTERS)
1399          ; *****
1400 004340 104001          TEST11: SCOPE
1401 004342 012777 000011 174532  MOV      #11, @DISPLAY      ; LOAD THE TEST NUMBER INTO THE DISPLAY
1402 004350 004767 007436          JSR     %7, CLRPAR        ; INITIALLY CLEAR ALL PARITY REGISTERS
1403 004354 005767 174170          TST     NOKT             ; KT11 PRESENT?
1404 004360 001004          BNE     1$              ; 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 004372 012700 000566          1$:  MOV      #MPRO, R0      ; SETUP TO GET ADDRESSES OF REGISTERS.PRESENT
1409 004376 012703 000770          MOV      #INDC0, R3
1410 004402 032710 000001          LUP11: BIT      #1, @R0
1411 004406 001003          BNE     LOOP11          ; IF THIS REG NOT PRESENT, SKIP
1412 004410 022713 000001          CMP      #1, (R3)      ; IS THIS A CORE PAR REG?
1413 004414 001407          BEQ     TEST11         ; YES, THEN TEST IT
1414                                     ; IF NOT CORE, SKIP THIS REGISTER
1415 004416 062700 000010          LOOP11: ADD     #10, R0
1416 004422 005723          TST     (R3)+
1417 004424 020027 000766          CMP     R0, #TREG

```

```

1418 004430 103764          BLC      LUP11
1419 004432 000443          BR       DONE11
1420 004434 004767 007466   TST11: JSR      %7,LOCATM
1421
1422 004440 032701 000001          BIT      #1,R1
1423 004444 001403          BEQ     .+10
1424 004446 104002          ERROR
1425
1426
1427 004450 000167 177742          JMP     LOOP11
1428 004454 010102          MOV     R1,R2
1429 004456 062702 010000          ADD     #10000,R2
1430 004462 012770 000004 000000          MOV     #WWP,@(R0)
1431 004470 011111          MOV     @R1,@R1
1432 004472 011212          MOV     @R2,@R2
1433 004474 005070 000000          CLR     @(R0)
1434 004500 021112          CMP     @R1,@R2
1435
1436 004502 005770 000000          TST     @(R0)
1437 004506 100401          BMI     .+4
1438 004510 104002          ERROR
1439
1440
1441 004512 032770 000100 000000          BIT     #BIT6,@(R0)
1442
1443
1444 004520 001001          BNE     .+4
1445 004522 104002          ERROR
1446
1447
1448
1449
1450 004524 005070 000000          CLR     @(R0)
1451 004530 005511          ADC     @R1
1452 004532 005512          ADC     @R2
1453 004534 005070 000000          CLR     @(R0)
1454 004540 000726          BR     LOOP11
1455 004542 005767 174002          DONE11: TST     NOKT
1456 004546 001002          BNE     .+6
1457 004550 005037 177572          CLR     @#SRO
1458
1459
1460
1461
1462
1463
1464
1465
1466 004554 104001          TEST12: SCOPE
1467 004556 012777 000012 174316          MOV     #12,@DISPLAY
1468 004564 004767 007222          JSR     %7,CLRPAR
1469 004570 005767 173754          TST     NOKT
1470 004574 001004          BNE     1$
1471 004576 004767 007110          JSR     %7,NRALL
1472 004602 004767 007254          JSR     %7,MAP1
1473

```

```

; BRANCH OUT IF ALL REGISTERS HAVE BEEN TESTED
; GET THE ADDRESS OF A MEMORY LOCATION
; CORRESPONDING TO THIS PARITY REGISTER
; ERROR RETURN INDICATED?
; BRANCH IF NOT
; NO MEMORY IN MAP CORRESPONDING TO
; THIS PARITY REGISTER. R0 POINTS
; TO THE ADDRESS OF THE PARITY REGISTER
; SETUP SECOND TEST ADDRESS LOCATION
; SET WRITE WRONG PARITY
; WRITE WRONG PARITY IN FIRST TEST LOCATION
; WRITE WRONG PARITY IN SECOND TEST LOCATION
; CLEAR PARITY REGISTER
; READ FIRST TEST LOCATION, AND
; THEN READ SECOND TEST LOCATION
; MAKE SURE PARITY ERROR SET
; PARITY ERROR NOT SET AFTER
; READING TWO LOCATIONS WHICH
; SHOULD HAVE BAD PARITY
; CHECK ERROR ADDRESS- IF THE LAST
; ADDRESS WAS RECORDED, BIT 6 WILL
; BE SET
; PARITY ERROR ADDRESS BITS INCORRECT
; R0 POINTS TO ADDRESS OF PARITY REGISTER
; R2 CONTAINS ADDRESS OF LAST BAD PARITY
; LOCATION REFERENCED (IF KT11 PRESENT
; ADDRESS IS VIRTUAL THRU KERNEL PAGE 1)
; CLEAR PARITY REGISTER
; CLEAR BAD PARITY
; CLEAR PARITY ERROR BIT
; TURN OFF KT11 IF PRESENT
; *****
; SHOW THAT IF AN INSTRUCTION DOING A DATIP GETS A PARITY ERROR,
; THE ORIGINAL DATA IS REWRITTEN IF ACTION ENABLE IS SET, AND IS
; ALTERED IF ACTION ENABLE IS CLEAR
; *****
; LOAD THE TEST NUMBER INTO THE DISPLAY
; KT11 PRESENT?
; NO- BRANCH
; YES, MAP KERNEL 0 TO BANK 0, RW
; MAP KERNEL 7 TO THE EXTERNAL BANK, RW
; SET KERNEL 1 RW AND TURN ON KT11

```

1474	004606	012700	000566		IS:	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		;BRANCH TO DONE IF ALL REGISTERS
1486									;HAVE BEEN TESTED
1487	004650	004767	007252		TST12:	JSR	%7,LOCATM		;LOCATE MEMORY CORRESPONDING TO THIS
1488									REGISTER
1489	004654	032701	000001			BIT	#1,R1		;ERROR RETURN INDICATED?
1490	004660	001403				BEQ	+.10		;NO- BRANCH
1491	004662	104002				ERROR			;NO MEMORY IN MAP CORRESPONDING TO
1492									;THIS REGISTER. R0 POINTS TO
1493									;THE ADDRESS OF THE PARITY REGISTER
1494	004664	000167	177742			JMP	LOOP12		
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									;RESTORE ORIGINAL DATA
1503	004720	104002				ERROR			;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 ALTERED 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									;ALTERED 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      2(R0)          ;CLEAR PARITY REGISTER
1531 004770 012737 000116 000114  MOV      2*PARVEC+2,2*PARVEC ;RESTORE TRAPCATCHER
1532 004776 005211          INC      2R1           ; SINCE AE IS CLEAR, INSTRUCTION SHOULD
1533          ; COMPLETE AND SHOULD CLEAR BAD PARITY
1534 005000 005070 000000          CLR      2(R0)          ;CLEAR PARITY ERROR BIT
1535 005004 022711 125253          CMP      2125253,2R1   ;CHECK DATA
1536 005010 001401          BEQ     .+4
1537 005012 104002          ERROR
1538          ;DATIP, DATO TO A LOCATION CONTAINING BAD
1539          ;PARITY WITHOUT AE SET LEFT INCORRECT
1540          ;DATA, R0 POINTS TO THE ADDRESS OF
1541          ;THE PARITY REGISTER. R1 CONTAINS THE
1542          ;ADDRESS OF THE TEST LOCATION (VIRTUAL
1543          ;THRU KERNEL PAGE 1 IF KT11 IS PRESENT)
1544 005014 005770 000000          TST     2(R0)          ;CHECK PARITY ERROR BIT
1545 005020 100001          BPL     .+4
1546 005022 104002          ERROR
1547          ;DATIP, DATO WITH AE CLEAR DID
1548          ;NOT CLEAR BAD PARITY IN LOCATION
1549          ;ADDRESSED. R0 POINTS TO THE ADDRESS
1550          ;OF THE PARITY REGISTER. R1 CONTAINS
1551          ;THE ADDRESS OF THE TEST LOCATION
1552          ;(VIRTUAL THRU KERNEL PAGE 1 IF KT11
1553          ;IS PRESENT)
1554 005024 005070 000000  CONT12: CLR      2(R0)          ;CLEAR PARITY REGISTER
1555 005030 005011          CLR      2R1           ;CLEAR LOCATION TO RESTORE GOOD PARITY
1556 005032 005070 000000          CLR      2(R0)          ;CLEAR PARITY ERROR IF SET
1557 005036 000675          BR     LOOP12         ;GO CHECK FOR ANOTHER PARITY
1558          ;REGISTER
1559 005040 012737 000116 000114  DONE12: MOV     2*PARVEC+2,2*PARVEC ;RESTORE TRAPCATCHER
1560 005046 005767 173476          TST     NOKT
1561 005052 001002          BNE     .+6
1562 005054 005037 177572          CLR     2*SR0
1563          ;TURN OFF KT11 IF PRESENT
1564          ;*****
1565          ;SHOW THAT IF AN INSTRUCTION DOING A DATI (BUT NO DATO TO THE SAME LOCATION)
1566          ;GETS A PARITY ERROR, THE ORIGINAL DATA IS UNALTERED, WHETHER OR NOT ACTION
1567          ;ENABLE IS SET
1568          ;*****
1569 005060 104001          TEST13: SCOPE
1570 005062 012777 000013 174012  MOV     213,2DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
1571 005070 004767 006716          JSR     27,CLRPAR
1572 005074 005767 173450          TST     NOKT
1573 005100 001004          BNE     1$
1574 005102 004767 006604          JSR     27,NRALL
1575 005106 004767 006750          JSR     27,MAP1
1576          ;KT11 PRESENT?
1577          ;NO- BRANCH
1578          ;YES, MAP KERNEL 0 TO BANK 0, KERNEL
1579          ;7 TO THE EXTERNAL BANK, AND KERNEL
1580          ;0, 1 AND 7 RW
1581 005112 012700 000566 1$: MOV     2MPRO,R0 ;SETUP TO GET ADDRESSES OF REGISTERS PRESENT
1582 005116 032710 000001  LUP13: BIT     21,2R0
1583 005122 001406          BEQ     TST13
1584          ;IF THIS REGISTER IS PRESENT, GO
1585          ;TEST IT
1586          ;*****
1587          ;*****
1588          ;*****
1589          ;*****
1590          ;*****
1591          ;*****
1592          ;*****
1593          ;*****
1594          ;*****
1595          ;*****
1596          ;*****
1597          ;*****
1598          ;*****
1599          ;*****
1600          ;*****
1601          ;*****
1602          ;*****
1603          ;*****
1604          ;*****
1605          ;*****
1606          ;*****
1607          ;*****
1608          ;*****
1609          ;*****
1610          ;*****
1611          ;*****
1612          ;*****
1613          ;*****
1614          ;*****
1615          ;*****
1616          ;*****
1617          ;*****
1618          ;*****
1619          ;*****
1620          ;*****
1621          ;*****
1622          ;*****
1623          ;*****
1624          ;*****
1625          ;*****
1626          ;*****
1627          ;*****
1628          ;*****
1629          ;*****
1630          ;*****
1631          ;*****
1632          ;*****
1633          ;*****
1634          ;*****
1635          ;*****
1636          ;*****
1637          ;*****
1638          ;*****
1639          ;*****
1640          ;*****
1641          ;*****
1642          ;*****
1643          ;*****
1644          ;*****
1645          ;*****
1646          ;*****
1647          ;*****
1648          ;*****
1649          ;*****
1650          ;*****
1651          ;*****
1652          ;*****
1653          ;*****
1654          ;*****
1655          ;*****
1656          ;*****
1657          ;*****
1658          ;*****
1659          ;*****
1660          ;*****
1661          ;*****
1662          ;*****
1663          ;*****
1664          ;*****
1665          ;*****
1666          ;*****
1667          ;*****
1668          ;*****
1669          ;*****
1670          ;*****
1671          ;*****
1672          ;*****
1673          ;*****
1674          ;*****
1675          ;*****
1676          ;*****
1677          ;*****
1678          ;*****
1679          ;*****
1680          ;*****
1681          ;*****
1682          ;*****
1683          ;*****
1684          ;*****
1685          ;*****
1686          ;*****
1687          ;*****
1688          ;*****
1689          ;*****
1690          ;*****
1691          ;*****
1692          ;*****
1693          ;*****
1694          ;*****
1695          ;*****
1696          ;*****
1697          ;*****
1698          ;*****
1699          ;*****
1700          ;*****
1701          ;*****
1702          ;*****
1703          ;*****
1704          ;*****
1705          ;*****
1706          ;*****
1707          ;*****
1708          ;*****
1709          ;*****
1710          ;*****
1711          ;*****
1712          ;*****
1713          ;*****
1714          ;*****
1715          ;*****
1716          ;*****
1717          ;*****
1718          ;*****
1719          ;*****
1720          ;*****
1721          ;*****
1722          ;*****
1723          ;*****
1724          ;*****
1725          ;*****
1726          ;*****
1727          ;*****
1728          ;*****
1729          ;*****
1730          ;*****
1731          ;*****
1732          ;*****
1733          ;*****
1734          ;*****
1735          ;*****
1736          ;*****
1737          ;*****
1738          ;*****
1739          ;*****
1740          ;*****
1741          ;*****
1742          ;*****
1743          ;*****
1744          ;*****
1745          ;*****
1746          ;*****
1747          ;*****
1748          ;*****
1749          ;*****
1750          ;*****
1751          ;*****
1752          ;*****
1753          ;*****
1754          ;*****
1755          ;*****
1756          ;*****
1757          ;*****
1758          ;*****
1759          ;*****
1760          ;*****
1761          ;*****
1762          ;*****
1763          ;*****
1764          ;*****
1765          ;*****
1766          ;*****
1767          ;*****
1768          ;*****
1769          ;*****
1770          ;*****
1771          ;*****
1772          ;*****
1773          ;*****
1774          ;*****
1775          ;*****
1776          ;*****
1777          ;*****
1778          ;*****
1779          ;*****
1780          ;*****
1781          ;*****
1782          ;*****
1783          ;*****
1784          ;*****
1785          ;*****
1786          ;*****
1787          ;*****
1788          ;*****
1789          ;*****
1790          ;*****
1791          ;*****
1792          ;*****
1793          ;*****
1794          ;*****
1795          ;*****
1796          ;*****
1797          ;*****
1798          ;*****
1799          ;*****
1800          ;*****
1801          ;*****
1802          ;*****
1803          ;*****
1804          ;*****
1805          ;*****
1806          ;*****
1807          ;*****
1808          ;*****
1809          ;*****
1810          ;*****
1811          ;*****
1812          ;*****
1813          ;*****
1814          ;*****
1815          ;*****
1816          ;*****
1817          ;*****
1818          ;*****
1819          ;*****
1820          ;*****
1821          ;*****
1822          ;*****
1823          ;*****
1824          ;*****
1825          ;*****
1826          ;*****
1827          ;*****
1828          ;*****
1829          ;*****
1830          ;*****
1831          ;*****
1832          ;*****
1833          ;*****
1834          ;*****
1835          ;*****
1836          ;*****
1837          ;*****
1838          ;*****
1839          ;*****
1840          ;*****
1841          ;*****
1842          ;*****
1843          ;*****
1844          ;*****
1845          ;*****
1846          ;*****
1847          ;*****
1848          ;*****
1849          ;*****
1850          ;*****
1851          ;*****
1852          ;*****
1853          ;*****
1854          ;*****
1855          ;*****
1856          ;*****
1857          ;*****
1858          ;*****
1859          ;*****
1860          ;*****
1861          ;*****
1862          ;*****
1863          ;*****
1864          ;*****
1865          ;*****
1866          ;*****
1867          ;*****
1868          ;*****
1869          ;*****
1870          ;*****
1871          ;*****
1872          ;*****
1873          ;*****
1874          ;*****
1875          ;*****
1876          ;*****
1877          ;*****
1878          ;*****
1879          ;*****
1880          ;*****
1881          ;*****
1882          ;*****
1883          ;*****
1884          ;*****
1885          ;*****
1886          ;*****
1887          ;*****
1888          ;*****
1889          ;*****
1890          ;*****
1891          ;*****
1892          ;*****
1893          ;*****
1894          ;*****
1895          ;*****
1896          ;*****
1897          ;*****
1898          ;*****
1899          ;*****
1900          ;*****
1901          ;*****
1902          ;*****
1903          ;*****
1904          ;*****
1905          ;*****
1906          ;*****
1907          ;*****
1908          ;*****
1909          ;*****
1910          ;*****
1911          ;*****
1912          ;*****
1913          ;*****
1914          ;*****
1915          ;*****
1916          ;*****
1917          ;*****
1918          ;*****
1919          ;*****
1920          ;*****
1921          ;*****
1922          ;*****
1923          ;*****
1924          ;*****
1925          ;*****
1926          ;*****
1927          ;*****
1928          ;*****
1929          ;*****
1930          ;*****
1931          ;*****
1932          ;*****
1933          ;*****
1934          ;*****
1935          ;*****
1936          ;*****
1937          ;*****
1938          ;*****
1939          ;*****
1940          ;*****
1941          ;*****
1942          ;*****
1943          ;*****
1944          ;*****
1945          ;*****
1946          ;*****
1947          ;*****
1948          ;*****
1949          ;*****
1950          ;*****
1951          ;*****
1952          ;*****
1953          ;*****
1954          ;*****
1955          ;*****
1956          ;*****
1957          ;*****
1958          ;*****
1959          ;*****
1960          ;*****
1961          ;*****
1962          ;*****
1963          ;*****
1964          ;*****
1965          ;*****
1966          ;*****
1967          ;*****
1968          ;*****
1969          ;*****
1970          ;*****
1971          ;*****
1972          ;*****
1973          ;*****
1974          ;*****
1975          ;*****
1976          ;*****
1977          ;*****
1978          ;*****
1979          ;*****
1980          ;*****
1981          ;*****
1982          ;*****
1983          ;*****
1984          ;*****
1985          ;*****
1986          ;*****
1987          ;*****
1988          ;*****
1989          ;*****
1990          ;*****
1991          ;*****
1992          ;*****
1993          ;*****
1994          ;*****
1995          ;*****
1996          ;*****
1997          ;*****
1998          ;*****
1999          ;*****
2000          ;*****

```

1586 005140 004767 006762
1587
1588
1589 005144 032701 000001
1590 005150 001403
1591 005152 104002
1592
1593
1594
1595 005154 000167 177744
1596 005160 012737 005214 000114
1597 005166 012770 000004 000000
1598 005174 012711 125252
1599 005200 012770 000001 000000
1600
1601 005206 005711
1602
1603 005210 104002
1604
1605
1606
1607
1608
1609 005212 000434
1610 005214 005070 000000
1611 005220 021127 125252
1612 005224 001401
1613 005226 104002
1614
1615
1616
1617
1618
1619
1620 005230 005770 000000
1621 005234 100401
1622 005236 104002
1623
1624
1625
1626
1627
1628 005240 022626
1629 005242 012770 000004 000000
1630 005250 012711 125252
1631 005254 005070 000000
1632 005260 012737 000116 000114
1633 005266 005711
1634
1635 005270 005070 000000
1636 005274 022711 125252
1637 005300 001401
1638 005302 104002
1639
1640
1641

TST13: JSR %7,LOCATM
BIT #1,R1
BEQ .+10
ERROR
JMP LOOP13
MOV #TRP13,#PARVEC
MOV #WHP,#1(R0)
MOV #125252,#R1
MOV #AE,#1(R0)
TST R1
ERROR
TRP13: BR CONT13
CLR R1
CMP #125252,R1
BEQ .+4
ERROR
TST R1(R0)
BNI .+4
ERROR
CMP (SP)+,(SP)+
MOV #WHP,R1(R0)
MOV #125252,R1
CLR R1(R0)
MOV #PARVEC+2,#PARVEC
TST R1
CLR R1(R0)
CMP #125252,R1
BEQ .+4
ERROR

LOCATE MEMORY CORRESPONDING TO THIS REGISTER, AND IF KT11 IS PRESENT MAP KERNEL 1 TO THAT MEMORY ERROR RETURN INDICATED? NO-BRANCH MAP INDICATES NO MEMORY WAS FOUND CORRESPONDING TO THIS REGISTER R0 POINTS TO THE ADDRESS OF THE PARITY REGISTER

SETUP PARITY TRAP RETURN SET WRITE WRONG PARITY WRITE WRONG PARITY SET ACTION ENABLE AND CLEAR WRITE WRONG PARITY DATA WITH ACTION ENABLE SET SHOULD ABORT LEAVING DATA UNCHANGED NO ABORT ON READING BAD PARITY WITH ACTION ENABLE SET, R0 POINTS TO THE ADDRESS OF THE PARITY REGISTER. R1 CONTAINS THE ADDRESS OF THE TEST LOCATION (VIRTUAL THRU KERNEL PAGE 1 IF KT11 IS PRESENT).

ABORT OCCURRED AS EXPECTED- CLEAR REGISTER ORIGINAL DATA RESTORED?

YES BRANCH DATA WHICH GOT A PARITY ERROR ALTERED THE CONTENTS OF THE LOCATION ADDRESSED. (R1 CONTAINS THE ADDRESS OF MEMORY BEING TESTED- IF KT11 IS PRESENT, ADDRESS IN R1 IS VIRTUAL) R0 POINTS TO THE ADDRESS OF THE PARITY REGISTER CHECK PARITY REGISTER BRANCH IF PARITY ERROR SET PARITY ERROR NOT SET AFTER READING DATA WITH BAD PARITY R0 POINTS TO THE ADDRESS OF THE PARITY REGISTER. R1 CONTAINS THE ADDRESS OF THE TEST LOCATION (VIRTUAL THRU KERNEL PAGE 1 IF KT11 PRESENT)

RESTORE STACK POINTER SET WRITE WRONG PARITY, CLEAR PARITY ERROR REWRITE DATA WITH WRONG PARITY CLEAR PARITY REGISTER RESTORE TRAPCATCHER DATA TO LOCATION WITH BAD PARITY AE NOT SET-INSTRUCTION SHOULD COMPLETE CLEAR PARITY ERROR BIT CHECK DATA

DATA TO LOCATION WITH BAD PARITY WITHOUT ACTION ENABLE SET LEFT INCORRECT DATA R1 CONTAINS THE ADDRESS OF THE TEST LOCATION (VIRTUAL THRU KERNEL PAGE 1

```

1642
1643
1644 005304 005070 000000 CONT13: CLR @ (R0)
1645 005310 005011 CLR @R1
1646 005312 005070 000000 CLR @ (R0)
1647 005316 000702 BR LOOP13
1648
1649 005320 012737 000116 000114 DONE13: MOV #PARVEC+2,@#PARVEC
1650 005326 005767 173216 TST NOKT
1651 005332 001002 BNE +6
1652 005334 005037 177572 CLR @#SRO
1653
1654
1655
1656
1657
1658
1659
1660 005340 104001
1661 005342 012777 000014 173532 MOV #14,@DISPLAY
1662 005350 005067 011102 CLR IMAX
1663 005354 004767 006432 JSR PC CLRPAR
1664 005360 012737 005734 000114 MOV #TRP14,@#PARVEC
1665 005366 022767 000002 173474 CMP #2,PHEML
1666 005374 103402 BLO +6
1667 005376 004767 005742 JSR PC,SMLSYS
1668 005402 032767 000002 173110 BIT #2,BITPT
1669 005410 001007 BNE 1$
1670 005412 012767 000004 173100 MOV #4,BITPT
1671 005420 012767 040000 173070 MOV #40000,ADRPT
1672 005426 000403 BR LOOP14
1673 005430 012767 020000 173060 1$: MOV #20000,ADRPT
1674 005436 036767 173056 173424 LOOP14: BIT BITPT,PHEML
1675 005444 001021 BNE TST14
1676 005446 032767 000002 173044 LUP14: BIT #2,BITPT
1677 005454 001001 EX: BNE EX
1678 005456 000402 BR +6
1679 005460 000167 000776 EX: JMP TEST16
1680 005464 062767 020000 173024 ADD #20000,ADRPT
1681 005472 006367 173022 ASL BITPT
1682 005476 022767 000200 173014 CMP #200,BITPT
1683 005504 003354 BGT LOOP14
1684
1685 005506 000443
1686 005510 012704 001036 TST14: BR DONE14
1687 005514 016767 172776 173006 MOV #PARPAT,R4
1688 005522 062767 020000 173000 MOV ADRPT,HIADR
1689 005530 016705 172762 ADD #20000,HIADR
1690 005534 005025 2$: CLR (5)+
1691 005536 020567 172766 CMP R5,HIADR
1692 005542 103774 BLO 2$
1693 005544 012701 000566 3$: MOV #MPRO,R1
1694 005550 032711 000001 BIT #1,@R1
1695 005554 001003 BNE +10
1696 005556 012771 000001 000000 MOV #AE,@ (R1)
1697 005564 062701 000010 ADD #10,R1

```

```

; IF KT11 IS PRESENT). R0 POINTS TO THE
; ADDRESS OF THE PARTIY REGISTER.
; CLEAR PARITY REGISTER
; CLEAR LOCATION
; CLEAR PARITY ERROR IF SET
; GO CHECK FOR ANOTHER PARITY
; REGISTER
; RESTORE TRAPCATCHER

```

```

; TURN OFF KT11 IF PRESENT

```

```

; *****
; CHECK PARITY MEMORY WITH SERIES OF PATTERNS FROM 4K TO 28K
; ENABLE PARITY TRAP
; *****

```

```

TEST14: SCOPE
; LOAD THE TEST NUMBER INTO THE DISPLAY
; DON'T ITERATE THE REST OF THE SUBTESTS
; CLEAR ALL PARITY REGISTERS
; SETUP PARITY TRAP RETURN
; TEST FOR AN 8K SYSTEM
; SYST > 8K
; SYST < OR = 8K
; TEST BANK INDICATOR FOR BANK 1
; INITIALIZE BANK INDICATOR TO BANK 2
; INITIALIZE MEMORY STARTING ADDRESS
; INITIALIZE BANK INDICATOR TO BANK 1
; DOES THIS 4K HAVE PARITY?
; YES, TEST IT
; TEST FOR BANK 1 INDICATOR IF SET GO TO TEST 16
; NO- UPDATE MEMORY ADDRESS
; UPDATE BIT POINTER
; THIS 28K DONE?
; NO, BRANCH TO SEE IF NEXT 4K
; SHOULD BE TESTED
; YES, EXIT
; INITIALIZE PATTERN POINTER
; SET UPPER LIMIT FOR THIS 4K
; INITIALLY CLEAR CORE BLOCK UNDER TEST
; INITIALIZE TO SET AE IN ALL REGISTERS
; SET ACTION ENABLE IF REGISTER IS PRESENT

```

```

1698 005570 020127 000766          CMP      R1,#TREG
1699 005574 103765          BLO      3$
1700 005576 004767 000024          4$: JSR      %7,TPCOPE          ;GO TO ROUTINE TO EXERCISE THIS 4K
;WITH THE CURRENT PATTERN
1701          TST      (4)+          ;UPDATE PATTERN
1702 005602 005724          TST      (4)          ;LAST PATTERN?
1703 005604 005714          BNE      4$          ;NO LOOP
1704 005606 001373          JSR      PC CLRPAR          ;YES CLEAR ALL PARITY REGISTERS
1705 005610 004767 006176          BR      LUP14          ;UPDATE AND CHECK NEXT 4K
1706 005614 000714          BR      LUP14          ;RESTORE TRAP CATCHER
1707 005616 012737 000116 000114 DONE14: MOV      #PARVEC+2,2#PARVEC
1708 005624 000473          BR      TEST15          ;GO TO NEXT TEST
1709
1710          ;ROUTINE TO WRITE AND CHECK EACH LOCATION IN 4K (STARTING AT ADDRESS
1711          ;IN ADRPT) WITH VALUE POINTED TO BY R4
1712 005626 016705 172664          TRPCORE: MOV     ADRPT,R5          ;SETUP R5 TO ADDRESS MEMORY
;LOCATION BEING CHECKED
1713          1$: MOV      (4),(5)          ;WRITE PATTERN INTO MEMORY
1714 005632 011415          MOV      (5),WAS          ;READ TEST LOCATION
1715 005634 011567 172676          MOV      (4),WAS          ;DATA OK?
1716 005640 021467 172672          CMP      .+4          ;YES- BRANCH
1717 005644 001401          BEQ      ERROR          ;DATA INCORRECT IN LOCATION WHOSE
1718 005646 104002          ERROR          ;ADDRESS IS IN R5. R4 POINTS TO THE
;DATA WRITTEN.
1719          TST      (5)+          ;UPDATE ADDRESS POINTER
1720          CMP      R5,HIADR          ;THIS 4K DONE?
1721 005650 005725          BLO      1$          ;NO BRANCH TO TEST NEXT LOCATION
1722 005652 020567 172652          CLR      TREG          ;YES, DID ANY PARITY ERRORS OCCUR
1723 005656 103765          ;WITHOUT TRAPPING?
1724 005660 005067 173102          2$: MOV      #MPRO,R1
1725          BIT      #1,(R1)
1726 005664 012701 000566          BNE      .+10
1727 005670 032711 000001          TST      2(R1)          ;YES- BRANCH
1728 005674 001003          BMI      3$
1729 005676 005771 000000          ADD      #10,R1
1730 005702 100406          CMP      R1,#TREG
1731 005704 062701 000010          BLO      2$
1732 005710 020127 000766          RTS      %7
1733 005714 103765          3$: MOV      2R1,TREG
1734 005716 000207          ERRORS
1735 005720 011167 173042          ;NO RETURN
1736 005724 104004          ;STORE ADDRESS OF REGISTER GETTING ERROR
;PARITY ERROR SET (WITH AE SET) AND
;NO TRAP OCCURRED. TREG CONTAINS
;ADDRESS OF PARITY REGISTER WHICH
;HAS ERROR BIT SET.
1737          ;SCAN FOR PARITY ERRORS AND PRINT
1738          ;18 BIT ADDRESSES OF THOSE FOUND.
1739          ;AFTER REPORTING EACH ERROR CLEAR IT
1740 005726 004767 006562          JSR      %7,PSCAN
1741
1742          RTS      %7
1743 005732 000207
1744          ;PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
1745          TRP14: CLR      TREG
1746 005734 005067 173026          MOV      #MPRO,R1          ;FIND PARITY REGISTER INDICATING PARITY ERROR
1747 005740 012701 000566          BIT      #1,(R1)
1748 005744 032711 000001          BNE      .+10
1749 005750 001003          TST      2(R1)          ;BRANCH IF PARITY ERROR SET
1750 005752 005771 000000          BMI      2$
1751 005756 100407          ADD      #10,R1
1752 005760 062701 000010          CMP      R1,#TREG
1753 005764 020127 000766

```



```

1754 005770 103765      BLO      15
1755 005772 104002      ERROR
1756 005774 000405      BR       35
1757
1758 005776 011167 172764    25:     MOV      @R1,REG
1759 006002 104004      ERRORS
1760
1761
1762
1763
1764 006004 004767 006504      JSR      %7,PSCAN
1765
1766 006010 022626    35:     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
1778 006024 005767 172520      TST     NOKT
1779 006030 001402      BEQ     .+6
1780 006032 000167 000424      JMP     TEST16
1781 006036 004767 005750      JSR     PC,CLRPAR
1782 006042 004767 005644      JSR     %7,NRALL
1783 006046 004767 006010      JSR     %7,MAP1
1784
1785 006052 012777 001600 173172    MOV      #1600,@KPAR1
1786 006060 005067 172470      CLR     LOWFLG
1787 006064 016767 173000 173002    MOV      PMEMH,PMEMX
1788 006072 012737 006402 000114      MOV     #TRAP15,@FARVEC
1789 006100 012767 000200 172412    MOV     #200,BITPT
1790 006106 036767 172406 172760    LOOP15: BIT    BITPT,PMEMX
1791 006114 001022      BNE     TST15
1792 006116 062777 000200 173126    LOP15:  ADD    #200,@KPAR1
1793 006124 006367 172370      ASL    BITPT
1794 006130 103366      BCC    LOOP15
1795 006132 005767 172416      TST    LOWFLG
1796 006136 001051      BNE    DONE15
1797 006140 005267 172410      INC    LOWFLG
1798 006144 016767 172722 172722    MOV     PMEMH,PMEMX
1799 006152 012767 000001 172340    MOV     #1,BITPT
1800 006160 000752      BR     LOOP15
1801 006162 012704 001036 172322    TST15: MOV     #PARPAT,R4
1802 006166 012767 020000      MOV     #20000,ADRPT
1803
1804 006174 012705 020000
1805 006200 005025    25:     MOV     #20000,R5
1806 006202 020527 040000      CLR     (5)+
1807 006206 103774      CMP     R5,#40000
1808 006210 012701 000566      BLO    25
1809

```

```

;PARITY TRAP TO 114 OCCURRED DURING
;TEST 14 BUT NO REGISTERS HAVE
;PARITY ERROR SET
;STORE ADDRESS OF REGISTER GETTING ERROR
;PARITY TRAP TO 114 OCCURRED DUE TO
;PARITY ERROR WHILE EXERCISING MEMORY
;R1 POINTS TO THE ADDRESS OF THE
;PARITY REGISTER HAVING PARITY ERROR
;BIT SET
;SCAN FOR BAD PARITY AND TYPE 18 BIT ADDRESSES
;OF LOCATIONS FOUND BAD
;RESTORE STACK POINTER
;RETURN (FROM JSR TO TPCORE) TO
;CHECK NEXT PATTERN

```

```

;*****
;CHECK PARITY MEMORY WITH SERIES OF PATTERNS ABOVE 28K
;ENABLE PARITY ERROR TRAPPING
;*****

```

```

†TEST15: SCOPE
;LOAD THE TEST NUMBER INTO THE DISPLAY
;KT11 PRESENT?
;YES- BRANCH
;NO, SKIP TEST
;CLEAR ALL PARITY REGISTERS
;MAP KERNEL 0 TO BANK 0,RW
;MAP KERNEL 7 TO THE EXTERNAL BANK
;SET KERNEL 1 RW AND TURN ON KT11
;MAP KERNEL PAGE 1 TO BEGINNING OF 28-32K
;CLEAR FLAG TO INDICATE CHECKING LOWER 64K
;SETUP PARITY TRAP RETURN
;INITIALIZE BIT POINTER
;DOES THIS 4K HAVE PARITY?
;YES, BRANCH TO TEST IT
;NO- MAP TO NEXT 4K
;UPDATE BIT POINTER
;BRANCH IF NOT DONE WITH 64K
;DONE WITH 128K?
;YES, BRANCH
;NO, SET FLAG INDICATING UPPER 64K
;SETUP PARITY MAP WORD
;SETUP BIT POINTER FOR UPPER 64K
;CONTINUE
;INITIALIZE PATTERN POINTER
;INITIALIZE VIRTUAL ADDRESS OF MEMORY
;BEING TESTED
;INITIALLY CLEAR CORE BLOCK UNDER TEST
;INITIALIZE TO SET ACTION ENABLE IN ALL
;PARITY REGISTERS

```

```

1810 006214 032711 000001 3$: BIT #1,R1
1811 006220 001003 BNE .+10
1812 006222 012771 000001 000000 MOV #AE,@(R1) ;SET ACTION ENABLE IF THIS REGISTER
1813 ;IS PRESENT
1814 006230 062701 000010 ADD #10,R1
1815 006234 020127 000766 CMP R1,#TREG
1816 006240 103765 BLO 3$
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 000114 DONE15: CLR @#SRO ;TURN OFF KT1 WHEN DONE
1824 00626E 012737 000116 MOV #PARVEC+2,@#PARVEC ;RESTORE 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 ;PCORX: NOP
1830 006276 000240 MOV #20000,R5 ;SETUP R5 TO POINT TO THE LOCATION
1831 ;UNDER TEST (VIRTUAL ADDRESS)
1832 006304 011415 1$ MOV (4),(5) ;WRITE PATTERN
1833 006306 011567 172224 MOV (5),WAS ;READ TEST LOCATION
1834 006312 021467 172220 CMP (4),WAS ;DATA OK?
1835 006316 001401 BEQ .+4 ;YES- BRANCH
1836 006320 104002 ERROR ;NO- DATA INCORRECT IN LOCATION WHOSE
1837 ;VIRTUAL ADDRESS IS IN R1 (GOES THRU
1838 ;KERNEL PAGE 1). R4 POINTS TO
1839 ;THE VALUE WRITTEN.
1840 006322 005725 TST (5)+ ;UPDATE ADDRESS POINTER
1841 006324 020527 040000 CMP R5,#40000 ;THIS 4K DONE?
1842 006330 103765 BLO 1$ ;NO BRANCH TO TEST NEXT LOCATION
1843 006332 005067 172430 CLR TREG ;YES, CHECK TO SEE IF ANY PARITY
1844 ;ERRORS OCCURRED WITHOUT TRAPPING
1845 006336 012701 000566 MOV #MPRO,R1
1846 006342 032711 000001 2$: BIT #1,(1) ;IS THIS PARITY REGISTER PRESENT?
1847 006346 001003 BNE .+10 ;NO, GET NEXT ONE
1848 006350 005771 000000 TST @(R1) ;YES- DID ERROR SET?
1849 006354 100406 BMI 3$ ;YES- BRANCH
1850 006356 062701 000010 ADD #10,R1 ;NO- GET NEXT REGISTER
1851 006362 020127 000766 CMP R1,#TREG
1852 006366 103765 BLO 2$
1853 006370 000207 RTS %7 ;NO ERRORS- EXIT
1854 006372 011167 172370 3$: MOV @R1,TREG ;STORE ADDRESS OF REGISTER GETTING ERROR
1855 006376 104004 ERRORS ;PARITY ERROR SET (AE ALREADY SET)
1856 ;AND NO TRAP OR TIMEOUT OCCURRED
1857 ;R1 POINTS TO THE ADDRESS OF THE
1858 ;PARITY REGISTER
1859 006400 000207 RTS %7
1860
1861 ;PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
1862 006402 005067 172360 TRP15: CLR TREG
1863 006406 012701 000566 MOV #MPRO,R1 ;LOCATE PARITY REGISTER INDICATING ERROR
1864 006412 032711 000001 1$: BIT #1,(R1)
1865 006416 001003 BNE .+10

```

1866	006420	005771	000000		TST	2(R1)		
1867	006424	100407			BMI	2\$; BRANCH IF PARITY ERROR IS SET
1868	006426	062701	000010		ADD	#10, R1		
1869	006432	020127	000766		CMP	R1, #TREG		
1870	006436	103765			BLO	1\$		
1871	006440	104002			ERROR			; TRAP TO 114 OCCURRED DURING TEST 15 BUT
1872								; NO PARITY REGISTERS HAVE PARITY ERROR SET
1873	006442	000405			BR	3\$		
1874	006444	011167	172316	2\$:	MOV	2R1, TREG		; STORE ADDRESS OF REGISTER GETTING ERROR
1875	006450	104004			ERRORS			; PARITY TRAP TO 114 OCCURRED DUE TO
1876								; PARITY ERROR WHILE EXERCISING MEMORY
1877								; "TREG" CONTAINS ADDRESS OF PARITY REGISTER
1878								; HAVING PARITY ERROR BIT SET
1879	006452	004767	006036		JSR	%7, PSCAN		; SCAN MEMORY FOR BAD PARITY AND PRINT 18
1880								; BIT ADDRESSES OF LOCATIONS FOUND
1881								; CLEAR BAD PARITY IN EACH AFTER
1882								; REPORTING IT
1883	006456	022626		3\$:	CMP	(SP)+, (SP)+		; RESTORE STACK POINTER
1884	006460	000207			RTS	%7		; RETURN (FROM JSR TO TPCORX) TO
1885								; TEST NEXT PATTERN
1886								
1887								
1888								
1889								
1890								
1891								
1892								
1893								
1894								
1895								
1896	006462	104001						
1897	006464	012777	000016	172410	TEST16: SCOPE			
1898	006472	012737	007076	000114	MOV	#16, 2DISPLAY		; LOAD THE TEST NUMBER INTO THE DISPLAY
1899	006500	032767	000002	172012	MOV	#TRP16, 2#PARVEC		; SET UP TRAP RETURN
1900	006506	001007			BIT	#2, BITPT		; TEST BANK INDICATOR FOR BANK 1
1901	006510	012767	000004	172002	BNE	LUP16		
1902	006516	012767	040000	171772	MOV	#4, BITPT		; INIT POINTER TO BANK 2
1903	006524	000403			MOV	#40000, ADRPT		
1904	006526	012767	020000	171762	BR	.+10		
1905	006534	036767	171760	172326	LUP16: MOV	#20000, ADRPT		
1906	006542	001021			1\$:	BIT	BITPT, #MEML	; DOES THIS 4K HAVE PARITY?
1907	006544	032767	000002	171746	BNE	3\$; YES, BRANCH TO TEST IT
1908	006552	001001			2\$:	BIT	#2, BITPT	; TEST FOR BANK 1 INDICATOR IF SET GO OUT
1909	006554	000402			BNE	.+4		
1910	006556	000167	001546		BR	.+6		
1911	006562	062767	020000	171726	JMP	OUT		
1912	006570	006367	171724		ADD	#20000, ADRPT		; NO, UPDATE MEMORY ADDRESS BY 4K
1913	006574	022767	000200	171716	ASL	BITPT		; UPDATE BIT POINTER
1914	006602	003354			CMP	#200, BITPT		; THIS 28K DONE?
1915	006604	000421			BGT	1\$; NO, CHECK NEXT 4K
1916	006606	016767	171704	171714	BR	DONE16		; YES, EXIT
1917	006614	052767	020000	171706	3\$:	MOV	ADRPT, HIADR	; SET UPPER LIMIT THIS 4K
1918	006622	004767	005164		ADD	#20000, HIADR		
1919	006626	016705	171664		JSR	%7, CLRPAR		; CLEAR ALL PARITY REGISTERS
1920	006632	005025			MOV	ADRPT, R5		
1921	006634	020567	171670		5\$:	CLR	(5)+	; CLEAR BANK UNDER TEST
					CMP	R5, HIADR		

```

;*****
; 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 000020
1923 006642 004767 000020 6$: BLO 5$
1924 006646 000736 000114 JSR %7,WWP16 ;GO WRITE WRONG PARITY IN EACH BYTE
1925 006650 004767 005136 BR 2$ ;UPDATE AND CHECK NEXT 4K
1926 006654 012737 000116 DONE16: JSR %7 CLRPAR ;CLEAR ALL PARITY REGISTERS IF DONE
1927 006662 000167 000472 MOV #PARVEC+2,@#PARVEC ;RESTORE TRAP CATCHER
1928 JMP TEST17 ;GO TO NEXT TEST
1929 ;WRITE WRONG PARITY TEST ROUTINE - TESTS EACH BYTE IN 4K
1930 ;USING SAME DATA VALUE WRITES AND CHECKS PARITY IN WRONG STATE
1931 ;AND THEN IN CORRECT STATE TO PROVE THAT PARITY BITS TOGGLE
1932 006666 016705 171624 WWP16: MOV ADRPT,R5 ;SET TEST ADDRESS POINTER
1933 006672 005067 171660 CLR ODDFLG ;INDICATE TESTING LOW BYTE
1934 006676 012767 125253 171630 MOV #125253,SHOBE ;STORE DATA FOR USE BY ERROR TYPEOUT ROUTINE
1935 006704 012715 125253 WWP16A: MOV #125253,@R5 ;INITIALIZE TEST LOCATION
1936 006710 012701 000566 MOV #MPRO,R1 ;SETUP TO LOAD PARITY REGISTERS
1937 006714 032711 000001 1$: BIT #1(1)
1938 006720 001003 000005 000000 BNE .+10
1939 006722 012771 000005 MOV #WWP+AE,@(R1) ;SET WRITE WRONG PARITY AND ACTION
1940 ;ENABLE IF THIS PARITY REGISTER
1941 ;IS PRESENT
1942 006730 062701 000010 ADD #10,R1
1943 006734 020127 000766 CMP R1,#TREG
1944 006740 103765 171610 BLO 1$ ;WRITING HIGH BYTE?
1945 006742 005767 000253 TST ODDFLG ;YES, BRANCH
1946 006746 100425 000566 BMI MOVB #253,@R5 ;NO, WRITE WRONG PARITY IN LOW BYTE
1947 006750 112715 000001 5$: MOV #MPRO,R1 ;THIS CODE CLEARS WWP BIT IN ALL
1948 006754 012701 000001 BIT #1(1) ;PARITY REGISTERS
1949 006760 032711 000004 BNE .+10
1950 006764 042771 000004 000000 BIC #WWP,@(R1)
1951 006774 062701 000010 ADD #10,R1
1952 007000 020127 000766 CMP R1,#TREG
1953 007004 103765 171544 BLO 5$+4 ;TESTING HIGH OR LOW BYTE?
1954 007006 005767 000377 TST ODDFLG ;BRANCH, IF HIGH BYTE
1955 007012 100407 000377 BMI 6$ ;DETECT WRONG PARITY WITH DATIP-
1956 007014 142715 000377 BICB #377,@R5 ;SHOULD TRAP TO TRP16 BEFORE DOING THE DATOB
1957 007020 000407 000252 000001 2$: BR 3$ ;WRITE WRONG PARITY IN HIGH BYTE
1958 007022 112765 000377 000001 6$: MOVB #252,1(P5)
1959 007030 000751 000377 000001 4$: BR 5$ ;DETECT WRONG PARITY WITH DATIP-
1960 007032 142765 000377 000001 3$: BICB #377,1(R5) ;SHOULD TRAP TO TRP16 BEFORE DOING THE DATOB
1961 007040 012701 000566 3$: MOV #MPRO,R1 ;NO TRAP OCCURRED- SETUP TO CLEAR
1962 007044 032711 000001 4$: BIT #1,@R1 ;AE AND WWP
1963 007050 001003 000005 000000 BNE .+10 ;CLEAR AE AND WWP IN ALL PARITY REGISTERS
1964 007052 042771 000010 BIC #WWP+AE,@(R1)
1965 007060 062701 000010 ADD #10,R1
1966 007064 020127 000766 CMP R1,#TREG
1967 007070 103765 4$
1968 007072 104002 ERROR ;ERROR, NO TRAP AFTER WRITING AND
1969 ;READING WRONG PARITY IN LOCATION
1970 ;WHOSE ADDRESS IS IN R5 (AE AND WWP
1971 ;WERE SET IN ALL PARITY REGISTERS)
1972 ;TESTING LOW BYTE IF ODDFLG IS POSITIVE
1973 ;TESTING HIGH BYTE IF ODDFLG IS NEGATIVE
1974
1975
1976
1977

```

;NOTE THAT AE AND WWP WERE CLEARED
;BEFORE TYPING THE ERROR PRINTOUT

```

1978
1979
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  TRP16A: BIT     #1,R1           ;WWP AND AE IN ALL REGISTERS
1985 007106 001003          BNE     .+10
1986 007110 042771 000005 000000  BIC     #AE+WWP,2(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$: BIT     #1,2R1          ;DOES THIS CONTROL EXIST?
1994 007150 001017          BNE     2$
1995 007152 005771 000000  TST     2(R1)            ;NO- BRANCH
1996 007156 100014          BPL     2$              ;YES- IS ERROR SET?
1997 007160 005767 171602  TST     TREG            ;NO- BRANCH
1998 007164 001431          BEQ     .+4            ;YES- WAS IT SET IN ANY OTHER REGISTER ALSO?
1999 007166 104002          ERROR          ;NO- BRANCH
2000
2001
2002 007170 036761 171324 000002  BIT     BITPT,2(R1)     ;ERROR SET IN MORE THAN ONE PARITY REGISTER
2003
2004 007176 001001          BNE     .+4            ;AFTER WRITING WRONG PARITY IN LOCATION
2005 007200 104002          ERROR          ;WHOSE ADDRESS IS IN RS
2006
2007
2008
2009
2010 007202 010304          MOV     R3,R4
2011 007204 011167 171556  MOV     2R1,TREG        ;DOES MAP INDICATE THIS PARITY REGISTER
2012 007210 062701 000010  ADD     #10,R1          ;CONTROLS THIS MEMORY?
2013 007214 005723          TST     (R3)+
2014 007216 020127 000766  CMP     R1,#TREG
2015 007222 103750          BLO     1$
2016
2017 007224 011567 171306  MOV     (5),WAS
2018 007230 022715 125253  CMP     #125253,2RS
2019 007234 001401          BEQ     .+4
2020 007236 104003          ERRORP
2021
2022
2023
2024
2025
2026
2027 007240 005767 171522  TST     TREG
2028 007244 001002          BNE     3$
2029 007246 104002          ERROR          ;STORE REGISTER ADDRESS
2030
2031
2032
2033

```

```

;BRANCH UNTIL ALL THE PARITY
;REGISTERS HAVE BEEN CHECKED
;SAVE DATA FROM LOCATION UNDER TEST
;DID BICB CHANGE DATA?
;NO, CONTINUE
;DATA WAS MODIFIED BY THE BICB WHICH
;GOT A PARITY ERROR TRAP- SINCE PARITY ERROR
;TRAP OCCURRED, CONTENTS SHOULD NOT HAVE
;BEEN MODIFIED. RS CONTAINS ADDRESS
;OF TEST LOCATION. "TREG" CONTAINS
;ADDRESS OF PARITY REGISTER SENSING
;ERROR
;WAS PARITY ERROR SET IN ANY REGISTERS?
;YES- BRANCH
;PARITY TRAP OCCURRED ON READING
;WRONG PARITY (WITH AE SET) BUT NO
;REGISTERS HAD PARITY ERROR BIT SET.
;RS CONTAINS THE ADDRESS OF THE
;TEST LOCATION.

```

```

2034 007250 000420
2035 007252 022714 000001    3$: BR 4$
2036 007256 001015          CMP #1,(R4)
2037 007260 017701 171502     MOV @TREG,R1
2038 007264 042701 170037     BIC #170037,R1
2039 007270 010502          MOV R5,R2
2040 007272 042702 003777     BIC #3777,R2
2041 007276 000302          SWAB
2042 007300 006302          ASL R2
2043 007302 006302          ASL R2
2044 007304 020102          CMP R1,R2
2045 007306 001401          BEQ .+4
2046 007310 104003          ERRORP
2047
2048
2049
2050
2051 007312 022626          4$: CMP (SP)+,(SP)+
2052 007314 011515          CNT16: MOV (5),(5)
2053 007316 005077 171444     CLR @TREG
2054 007322 005715          TST @R5
2055 007324 005777 171436     TST @TREG
2056 007330 100001          BPL .+4
2057 007332 104002          ERROR
2058
2059
2060
2061
2062
2063 007334 005167 171216     CN16: COM 00DFLG
2064 007340 100401          BMI .+4
2065 007342 005725          TST (5)+
2066 007344 020567 171160     CMP R5,HIADR
2067 007350 103401          BLO 1$
2068 007352 000207          RTS
2069 007354 000167 177324     1$: JMP WWP16A
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080 007360 104001
2081 007362 004767 005032
2082 007366 012777 000017 171506
2083 007374 012777 000017 171500
2084 007402 005767 171142
2085 007406 001402
2086 007410 000167 000636
2087 007414 004767 004372
2088 007420 012737 007762 000114
2089 007426 012767 000200 171064

```

;GET PARITY REGISTER CONTENTS
;MASK OFF ALL BUT ERROR ADDRESS BITS
;GET ADDRESS OF LOCATION UNDER TEST
;POSITION BITS IN R2
;PARITY ERROR ADDRESS BITS CORRECT?
;ERROR ADDRESS BITS (PARITY REGISTER
;BITS 11-5) ARE INCORRECT. R5 CONTAINS
;THE ADDRESS OF THE TEST LOCATION.
;"TREG" CONTAINS THE ADDRESS OF THE
;PARITY REGISTER DETECTING THE ERROR
;RESTORE STACK POINTER
;RESTORE TEST LOCATION TO FIX BAD PARITY
;CLEAR ERROR BIT IN 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
;R5 CONTAINS ADDRESS OF THE TEST
;LOCATION. "TREG" CONTAINS THE ADDRESS
;OF THE PARITY REGISTER DETECTING
;THE ERROR
;TOGGLE BYTE INDICATOR
;BRANCH IF READY TO TEST HIGH BYTE
;UPDATE ADDRESS POINTER
;THIS 4K DONE?
;NO, TEST NEXT LOCATION
;RETURN TO TEST NEXT BANK

```

*****
;FORCE WRONG PARITY IN EACH BYTE OF PARITY MEMORY ABOVE 28K
;WRITE WRONG PARITY AND READ IT WITH ACTION ENABLE SET, MAKING SURE
;THAT A TRAP OCCURS, THEN WRITE AND READ THE SAME LOCATION WITH GOOD PARITY
;(USING SAME DATA) TO SHOW THAT THE PARITY BIT TOGGLES. MAKE SURE THAT
;THE ERROR ADDRESS BITS (PARITY REGISTER BITS 5-11) ARE CORRECT.
*****
TEST17: SCOPE
        JSR  PC,RSTLDR
        MOV  #17,@DISPLAY
        MOV  #17,@DISPLAY
        NOK  NOK↑
        BEQ  .+6
        JMP  XFR1
        JSR  PC,CLAPAR
        MOV  #TRP17,@PARVEC
        MOV  #200,BITPT
;LOAD THE TEST NUMBER INTO THE DISPLAY
;KT11 PRESENT?
;YES, BRANCH
;NO, SKIP TO NEXT TEST
;CLEAR ALL PARITY REGISTERS
;SETUP FOR PARITY TRAP
;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 K111 AND MAKE KERNEL 1 RW
2094	007444	012777	001600	171600	MOV	#1600,@KPAR1			: INITIALIZE KERNEL PAGE 1 TO 28K
2095	007452	005067	171076		CLR	LOWFLG			: CLEAR FLAG TO INDICATE TESTING LOWER 64K
2096	007456	016767	171406	171410	MOV	PMEMH,PMEMX			
2097	007464	012767	000002	000366	MOV	#2,INDX17			: SETUP OFFSET TO CHECK LOWER 64K OF MAPS
2098	007472	036767	171022	171374	1\$: BIT	BITPT,PMEMX			: DOES THIS 4K HAVE PARITY?
2099	007500	001025			BNE	3\$: YES, BRANCH TO TEST IT
2100	007502	062777	000200	171542	2\$: ADD	#200,@KPAR1			: NO, MAP TO NEXT 4K
2101	007510	006367	171004		ASL	BITPT			: UPDATE BIT POINTER
2102	007514	103366			BCC	1\$: GO CHECK TO SEE IF THIS 4K HAS PARITY
2103	007516	005767	171032		TST	LOWFLG			: END OF 128K?
2104	007522	001025			BNE	DONE17			: YES, EXIT
2105	007524	005267	171024		INC	LOWFLG			: NO, SET FLAG TO INDICATE SHIFT TO
2106	007530	016767	171336	171336	MOV	PMEMH,PMEMX			: HIGH 64K AND CHANGE MAPS
2107	007536	012767	000001	170754	MOV	#1,BITPT			
2108	007544	012767	000004	000306	MOV	#4,INDX17			: SETUP OFFSET TO CHECK UPPER 64K OF MAPS
2109	007552	000747			BR	1\$			
2110	007554	012705	020000		3\$: MOV	#20000,R5			: CLEAR BANK UNDER TEST
2111	007560	005025			5\$: CLR	(5)+			
2112	007562	020527	040000		CMP	R5,#40000			
2113	007566	103774			5\$: BLO				
2114	007570	004767	000020		6\$: JSR	%7,WWP17			: 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 K111 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		WWP17: 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,SHD8E			: STORE DATA FOR USE BY ERROR TYPEOUT ROUTINE
2124	007632	012715	125253		WWP17A: 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	#WWP+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			1\$: BLO				: NO- LOOP
2133	007670	005767	170662		TST	ODDFLG			: YES- TESTING HIGH BYTE?
2134	007674	100405			BMI	2\$: YES, BRANCH
2135	007676	112715	000253		MOV8	#253,@R5			: NO, WRITE WRONG PARITY IN LOW BYTE
2136	007702	142715	000377		BIC8	#377,@R5			: DETECT WRONG PARITY WITH DATIP-
2137									: SHOULD TRAP TO TRP17 BEFORE DOING THE DATOB
2138	007706	000406			BR	3\$			
2139	007710	112765	000252	000001	2\$: MOV8	#252,(R5)			: WRITE WRONG PARITY IN HIGH BYTE
2140	007716	142765	000377	000001	BIC8	#377,(R5)			: DETECT WRONG PARITY WITH DATIP
2141									: SHOULD TRAP TO TRP17 BEFORE DOING THE DATOB
2142	007724	012701	000566		3\$: MOV	#MPRO,R1			: IF NO TRAP, CLEAR AE AND WWP IN ALL
2143	007730	032711	000001		4\$: BIT	#1,@R1			: PARITY REGISTERS
2144	007734	001003			BNE	+10			
2145	007736	042771	000005	000000	BIC	#AE+WWP,@(R1)			

```

2146 007744 062701 000010
2147 007750 020127 000766
2148 007754 103765
2149 007756 104002
2150
2151
2152
2153
2154
2155
2156
2157 007760 000512
2158
2159
2160 007762 012701 000566
2161 007766 032711 000001
2162 007772 001003
2163 007774 042771 000005 000000
2164 010002 062701 000010
2165 010006 020127 000766
2166 010012 103765
2167 010014 012701 000566
2168 010020 012703 000770
2169 010024 005067 170736
2170 010030 032711 000001
2171 010034 001017
2172 010036 005771 000000
2173 010042 100014
2174 010044 005767 170716
2175 010050 001401
2176 010052 104002
2177
2178
2179 010054 036761 170440 000002
2180
2181 010060 010060
2182 010062 001031
2183 010064 104002
2184
2185
2186
2187
2188
2189 010066 011167 170674
2190 010072 010304
2191 010074 062701 000010
2192 010100 005723
2193 010102 020127 000766
2194 010106 103750
2195
2196 010110 011567 170422
2197 010114 022715 125253
2198 010120 001401
2199 010122 104002
2200
2201

```

```

ADD #10,R1
CMP R1,#TREG
BLO 45
ERROR

BR CNT17

;WHEN WRONG PARITY DATA IS READ BACK, SHOULD ENTER HERE VIA TRAP TO 114
TRAP17: MOV #MPRO,R1
TRAP17A: BIT #1,R1
BNE .+10
BIC #AE+WWP,2(R1)
ADD #10,R1
CMP R1,#TREG
BLO TRAP17A
MOV #MPRO,R1
MOV #INDC0,R3
CLR TREG
15: BIT #1,R1
BNE 25
TST 2(R1)
BPL 25
TST TREG
BEQ .+4
ERROR

BIT BITPT,2(R1)

INDX17=-2
BNE .+4
ERROR

MOV 2R1,TREG
MOV R3,R4
25: ADD #10,R1
TST (R3)+
CMP R1,#TREG
BLO 15

MOV (5),WAS
CMP #125253,2R5
BEQ .+4
ERROR

```

```

;NO TRAP AFTER WRITING AND READING
;WRONG PARITY WITH AE SET- VIRTUAL
;ADDRESS OF LOCATION IS IN R5
;(MAPPED THRU KERNEL PAGE 1)
;WROTE LOW BYTE IF OODFLG IS POSITIVE
;WROTE HIGH BYTE IF IT IS NEGATIVE
;NOTE THAT WWP AND AE WERE CLEARED
;BEFORE ERROR PRINTOUT

;PARITY TRAP OCCURRED- BEFORE CHECKING
;IT, CLEAR WWP AND AE IN ALL REGISTERS

;FIND REGISTER THAT SENSED THE ERROR
;SETUP PTR TO INDICATOR

;DOES THIS CONTROL EXIST?
;NO- BRANCH
;YES- IS ERROR SET?
;NO- BRANCH
;YES- WAS IT SET IN ANY OTHER REGISTER ALSO?
;NO- BRANCH
;ERROR SET IN MORE THAN ONE PARITY REGISTER
;AFTER READING WRONG PARITY IN LOCATION
;WHOSE VIRTUAL ADDRESS IS IN R5
;DOES MAP INDICATE THAT THIS REGISTER
;CONTROLS THIS MEMORY?

;YES- BRANCH
;PARITY REGISTER RESPONDED TO MEMORY
;NOT INCLUDED IN ITS MAP. R1 POINTS TO
;THE PARITY REGISTER'S ADDRESS. R5
;CONTAINS VIRTUAL ADDRESS OF THE LOCATION
;BEING TESTED (MAPPED THRU KERNEL
;PAGE 1)
;STORE REGISTER ADDRESS
;STORE PTR TO INDICATOR

;INCREMENT PTR TO INDICATOR

;LOOP UNTIL ALL THE PARITY REGISTERS
;HAVE BEEN CHECKED
;SAVE CONTENTS OF LOCATION UNDER TEST
;DID BICB CHANGE DATA?
;NO, CONTINUE
;DATA WAS MODIFIED BY THE BICB WHICH
;GOT A PARITY ERROR TRAP- SINCE PARITY
;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     ' $
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 DION'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 BE 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 ;XXXXX
2263 010262 000167 175052 JMP TEST14 ;XXXX
2264 010265 005737 000042 TST #04E
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 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 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$: JMP TEST20+20010 ;GO TO TEST 20 IN BANK 1
2294
2295
2296
2297 ;*****
2298 ;IF FIRST 4K IS PARITY MEMORY, CHECK IT WITH A SERIES OF PATTERNS
2299 ;THIS SUBTEST IS RUN IN BANK 1 (20000 ABOVE THE ADDRESSES IN THE LISTING)
2300 ;*****
2301 010452 013746 177776 TEST20: MOV #PS, -(SP) ;THESE 2 LINES DO THE SAME AS A SCOPE WITHOUT
2302 010456 004767 005674 JSR PC,SCOPEC ;USING AN EMT
2303 010462 012777 000020 170412 MOV #20,DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY
2304 010470 004767 003316 JSR %7,CLRPAR ;CLEAR ALL PARITY REGISTERS
2305 010474 012704 021036 MOV #PARPAT+20000,R4 ;INITIALIZE PATTERN POINTER
2306 010500 005005 CLR R5
2307 010502 005025 1$: 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,2(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,CBK0	;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+2,2#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$: MOV (4),(5) ;WRITE PATTERN
2333 010604 021415 CMP (4),(5) ;DATA OK?
2334 010606 001404 BEQ ;YES- BRANCH
2335 010610 013746 MOV @,S-(SP) ;SETUP TO DO ERROR CALL VIA JSR
2336 010614 004767 JSR PC,ERR ;ERROR- DATA INCORRECT IN LOCATION
2337 ;WHOSE ADDRESS IS IN R5. R4 POINTS
2338 ;TO THE VALUE WRITTEN.
2339 010620 005725 TST (5)+ ;UPDATE ADDRESS POINTER
2340 010622 020527 000114 CMP R5,#114 ;DON'T CHANGE CONTENTS OF 114 AND 116
2341 010626 001002 B NE +6
2342 010630 062705 000004 AND #4,R5
2343 010634 020527 020000 CMP R5,#20000 ;HAS THE WHOLE BANK BEEN TESTED WITH
2344 ;THIS PATTERN?
2345 010640 103760 BLO 1$ ;NO, BRANCH TO TEST NEXT LOCATION
2346 010642 005067 170120 CLR TREG ;YES, DID ANY PARITY ERROR BITS SET?
2347 010646 012701 020566 MOV #MPRO+20000,R1
2348 010652 032711 000001 2$: BIT #1 (1)
2349 010656 001003 BNE +10
2350 010660 005771 000000 TST @ (R1)
2351 010664 100406 BMI 3$ ;YES- BRANCH
2352 010666 062701 000010 ADD #10,R1
2353 010672 020127 020766 CMP R1,#TREG+20000
2354 010676 103765 BLO 2$
2355 010700 000207 RTS %7 ;NO- RETURN
2356 010702 013746 177776 3$: MOV @#PS-(SP) ;SETUP TO DO ERROR CALL VIA JSR
2357 010706 004767 JSR PC,ERR ;ERROR- PARITY ERROR BIT SET AND NO
2358 ;PARITY TRAP OCCURRED
2359 ; (AE WAS SET) - R1 POINTS TO ADDRESS
2360 ; OF PARITY REGISTER
2361 010712 022626 177644 CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
2362 010714 000167 JMP DONE20
2363
2364 ;PARITY TRAP SERVICE (NO TRAPS TO 114 SHOULD OCCUR IN THIS SUBTEST)
2365 010720 005067 170042 TRP20: CLR TREG
2366 010724 012701 020566 MOV #MPRO+20000,R1 ;FIND THE REGISTER RECORDING A PARITY ERROR
2367 010730 032711 000001 1$: BIT #1 (R1)
2368 010734 001003 BNE +10
2369 010736 005771 000000 TST @ (R1)
2370 010742 100412 BMI 2$ ;BRANCH IF ERROR IS SET
2371 010744 062701 000010 ADD #10,R1
2372 010750 020127 020766 CMP R1,#TREG+20000
2373 010754 103765 BLO 1$
2374 010756 013746 177776 MOV @#PS-(SP) ;SETUP TO DO ERROR CALL VIA A JSR
2375 010762 004767 JSR PC,ERR ;ERROR- PARITY TRAP TO 114 OCCURRED DURING
2376 ;TEST 20 BUT NO REGISTERS HAVE PARITY
2377 ;ERROR SET
2378 010766 000430 BR 5$ ;SAVE CONTENTS OF PARITY REGISTER
2379 010770 017102 000000 2$: MOV @ (R1),R2 ;INITIALIZE ADDRESS POINTER
2380 010774 005003 CLR R3 ;CLEAR PARITY REGISTER
2381 010776 005071 000000 3$: CLR @ (R1)
2382 011002 005713 TST @R3 ;READ LOCATION
2383 011004 005771 000000 TST @ (R1) ;PARITY ERROR SET?
2384 011010 100413 BMI 4$ ;YES- BRANCH

```

```

2384 011012 005723          TST      (R3)+          ;MOVE POINTER
2385 011014 020327 020000  CMP      R3,#20000
2386 011020 103766          BLO     3$
2387 011022 010271 000000  MOV      R2,R(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
2409
2410
2411
2412
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 170010  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      RS
2418 011100 005025          1$:     CLR      (RS)+      ;CLEAR PARITY ERRORS IN BANK 0
2419 011102 020527 020000  CMP      RS,#20000
2420 011106 103774          BLO     1$
2421
2422
2423
2424
2425 011110 005005          ;WRITE WRONG PARITY TEST ROUTINE
2426 011112 005067 167440  ;USING SAME DATA VALUE, WRITES AND CHECKS PARITY IN WRONG STATE
2427 011116 012767 125253 167410  ;AND THEN IN CORRECT STATE TO PROVE THAT PARITY BITS TOGGLE
2428 011124 012715 125253  WWP21:  CLR      RS          ;SET TEST ADDRESS POINTER
2429 011130 012701 J20566  WWP21A: CLR      ODDFLG    ;CLEAR FLAG TO INDICATE TESTING LOW BYTE
2430                                     ;STORE DATA FOR USE BY ERROR TYPEOUT ROUTINE
2431 011134 032711 000001 1$:     MOV      #125253,%R5 ;INITIALIZE LOCATION
2432 011140 001003          MOV      #MPRO+20000,R1 ;SETUP TO SET WRITE WRONG PARITY
2433 011142 012771 000004 000000  MOV      #10,%R1       ;IN ALL REGISTERS
2434 011150 062701 000010  ADD      #10,R1
2435 011154 020127 020766  CMP      R1,%TREG+20000
2436 011160 103765          BLO     1$
2437 011162 005767 167370  TST      ODDFLG        ;TESTING HIGH BYTE?
2438 011166 100404          BMI     2$            ;YES, BRANCH
2439 011170 112715 000253  MOV      #253,%R5     ;NO, WRITE WRONG PARITY IN LOW BYTE

```

2440	011174	005715			TST	(R5)
2441	011176	000405			BR	3\$
2442	011200	112765	000252	000001	2\$: MOV	#252,1(P5)
2443	011206	105765	000001		TST	1(R5)
2444	011212	012701	020566		3\$: MOV	#MPRO+20000,R1
2445	011216	012703	000770		MOV	#INDCO,R3
2446	011222	032711	000001		4\$: BIT	#1,R1
2447	011226	001003			BNE	+10
2448	011230	042771	000004	000000	BIC	#WMP,2(R1)
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
2453	011254	005067	167506		CLR	TREG
2454	011260	032711	000001	LOG21.	BIT	#1,R1
2455	011264	001024			BNE	5\$
2456	011266	005771	000000		TST	2(R1)
2457	011272	100021			BPL	5\$
2458	011274	005767	167466		TST	TREG
2459	011300	001404			BEQ	+12
2460	011302	013746	177776		MOV	2#PS -(SP)
2461	011306	004767	002250		JSR	PC,ERR
2462						
2463						
2464	011312	032761	000001	000002	BIT	#1,2(R1)
2465						
2466	011320	001004			BNE	+12
2467	011322	013746	177776		MOV	2#PS -(SP)
2468	011326	004767	002230		JSR	PC,ERR
2469						
2470						
2471						
2472						
2473	011332	011167	167430		MOV	2R1,TREG
2474	011336	062701	000010	5\$:	ADD	#10,R1
2475	011342	005723			TST	(R3)+
2476	011344	020127	020766		CMP	R1,#TREG+20000
2477	011350	103743			BLO	LOOP21
2478						
2479	011352	005767	167410		TST	TREG
2480	011356	001005			BNE	6\$
2481	011360	013746	177776		MOV	2#PS -(SP)
2482	011364	004767	002172		JSR	PC,ERR
2483						
2484						
2485	011370	000423			BR	7\$
2486	011372	022713	000001	6\$:	CMP	#1,(R3)
2487	011376	001020			BNE	7\$
2488						
2489	011400	017701	167362		MOV	2TREG,R1
2490	011404	042701	170037		BIC	#170037,R1
2491	011410	010502			MOV	R5,R2
2492	011412	042702	003777		BIC	#3777,R2
2493	011416	000302			SWAB	R2
2494	011420	006302			ASL	R2
2495	011422	006302			ASL	R2

```

;DETECT WRONG PARITY WITH DATI
;WRITE WRONG PARITY IN HIGH BYTE
;DETECT WRONG PARITY WITH DATI
;SETUP TO CLEAR WWP IN ALL PARITY REGISTERS
;POINTER TO INDICATOR

;CLEAR WWP IN ALL PARITY REGISTERS

;CHECK TO SEE IF ANY REGISTER DETECTED
;THE PARITY ERROR

;CHECK PARITY REGISTER
;BRANCH IF PARITY ERROR IS NOT SET
;WAS PARITY ERROR SET IN ANY OTHER REGISTER?
;NO- BRANCH
;SETUP TO DO ERROR CALL VIA A JSR
;ERROR- PARITY ERROR SET IN MORE THAN
;ONE REGISTER AFTER WRITING WRONG PARITY
;IN LOCATION WHOSE ADDRESS IS IN R5
;DOES MAP INDICATE THAT THIS REGISTER
;CONTROLS THIS MEMORY?
;YES, BRANCH
;NO- SETUP TO DO ERROR CALL VIA A JSR
;ERROR- PARITY REGISTER RESPONDED
;TO MEMORY WHICH IS NOT IN ITS MAP
;R1 POINTS TO PARITY REGISTER'S ADDRESS.
;R5 CONTAINS ADDRESS OF LOCATION BEING
;TESTED
;STORE REGISTER ADDRESS

;BRANCH UNTIL ALL REGISTERS HAVE BEEN
;CHECKED
;WAS PARITY ERROR SET IN ANY REGISTER?
;YES- BRANCH
;NO- SETUP TO DO ERROR CALL VIA A JSR
;ERROR- NO REGISTER HAS PARITY ERROR
;SET AFTER READING WRONG PARITY IN LOCATION
;WHOSE ADDRESS IS IN R5

;IS THIS A CORE PAR REG?
;NO- BRANCH(MOS PAR REG DOES NOT
;STORE ADDR BITS OF PAR ERROR)
;GET PARITY REGISTER CONTENTS
;MASK ALL BUT ERROR ADDRESS BITS
;GET ADDRESS OF LOCATION UNDER TEST
;POSITION BITS IN R2

```

```

2496 011424 020102
2497 011426 001404
2498 011430 013746 177776
2499 011434 004767 002122
2500
2501
2502
2503 011440 011515
2504 011442 005077 167320
2505 011446 005715
2506 011450 005777 167312
2507 011454 100004
250E 011456 013746 177776
250F 011462 004767 002074
2510
2511
2512
2513
2514 011466 005167 167064
2515 011472 100401
2516 011474 005725
2517 011476 020527 020000
2518 011502 103610
2519 011504 004767 002302
2520 011510 013746 177776
2521 011514 004767 004636
2522
2523
2524 011520 012700 017770
2525 011524 005001
2526 011526 016111 020000
2527 011532 005721
2528 011534 005300
2529 011536 001373
2530 011540 162706 020000
2531 011544 162737 020000 013632
2532 011552 162737 020000 013634
2533 011560 162737 020000 013644
2534 011566 162737 020000 013646
2535 011574 162737 020000 013656
2536 011602 022737 177570 001100
2537 011610 001403
2538 011612 162737 020000 001100
2539
2540 011620 000167 160000
2541
2542
2543
2544
2545
2546
2547 011624 012737 000116 000114
2548 011632 012706 000510
2549 011636 004767 002150
2550 011642 005237 011752
2551 011646 004567 003544

```

```

CMP R1,R2
BEQ 7$
MOV @#PS -(SP)
JSR PC,ERR

7$: MOV @R5 @R5
CLR @TREG
TST @R5
TST @TREG
BPL +12
MOV @#PS -(SP)
JSR PC,ERR

COM ODDFLG
BMI +4
TST (R5)+
CMP R5,#20000
BLO WWP21A
JSR PC,CLRPAR
MOV @#PS -(SP)
JSR PC,SCOPEC

;COPY SECOND 4K BANK BACK TO FIRST 4K AND RETURN TO FIRST 4K BANK
XFR2: MOV #17770,R0
CLR R1
1$: MOV 20000(R1),@R1
TST (R1)+
DEC R0
BNE 1$
SUB #20000,SP
SUB #20000,@#ERRA1
SUB #20000,@#ERRA2
SUB #20000,@#ERRA3
SUB #20000,@#ERRA4
SUB #20000,@#ERRA5
CMP #177570,@#SWR
BEQ 2$
SUB #20000,@#SWR

2$: JMP DONE-20000

;*****
;AT THIS POINT EXECUTION RETURNS TO BANK 0
;*****
DONE: MOV @#PARVEC+2,@#PARVEC
MOV @STKPT,SP
JSR %? CLRPAR
INC @#PASCNT
JSR R5,OACNV

```

```

;PARITY ERROR ADDRESS BITS CORRECT?
;BRANCH IF YES
;NO- SETUP TO DO ERROR CALL VIA A JSR
;ERROR- ADDRESS BITS (PARITY REGISTER
;BITS 5-11) INCORRECT- ADDRESS OF PARITY
;REGISTER IS CONTAINED IN LOCATION "TREG"
;ADDRESS OF TEST LOCATION IS IN R5
;RESTORE TEST LOCATION TO FIX BAD PARITY
;CLEAR ERROR BIT IN PARITY REGISTER
;READ LOCATION TO SET IF PARITY IS GOOD
;CHECK PARITY ERROR BIT
;BRANCH IF NOT SET
;SETUP TO DO ERROR CALL VIA A JSR
;ERROR- WRITING LOCATION WITH WRITE
;WRONG PARITY CLEAR DIDN'T CLEAR BAD
;PARITY (ADDRESS OF LOCATION IS IN R5)
;"TREG" +20000 CONTAINS THE ADDRESS
;OF THE PARITY REGISTER
;TOGGLE BYTE INDICATOR
;BRANCH IF HIGH BYTE NOT YET TESTED
;UPDATE ADDRESS POINTER
;THIS 4K DONE?
;LOOP TILL ALL 4K HAS BEEN TESTED
;CLEAR ALL PARITY REGISTERS
;SETUP TO CALL SCOPE VIA JSR
;SCOPE

;RO IS USED AS A COUNTER
;R1 POINTS TO THE CURRENT LOCATION
;COPY BANK 1 TO BANK 0

;RESTORE STACK POINTER

;DOES THE CONSOLE HAVE A SWITCH REG. ?
;IF SO THEN GO TO 2$
;OTHERWISE RESTORE THE ADDRESS OF SOFTWARE
;SWITCH REGISTER
;RETURN TO BANK 0

;RESTORE TRAPCATCHER
;REINITIALIZE STACK POINTER
;CLEAR ALL PARITY REGISTERS
;KEEP TRACK OF PASSES COMPLETED

```



```

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$:    CMP      #7600, TBANK  ;EXTERNAL BANK YET?
2611 012132 003345          BGT      2$           ;NO, NOT YET
2612 012134 005037 177572          CLR      @#SRO       ;YES- DISABLE KT11
2613 012140 012737 000006 000004      MOV      #6, @#4     ;RESTORE TRAPCATCHER
2614 012146 000207          RTS      %7          ;RETURN
2615 012150 046777 166344 166370 5$:    BIC      BITPT, @MEMUT ;TIMEOUT OCCURRED-CLEAR BIT TO INDICATE
2616                                ;4K BLOCK NOT PRESENT
2617 012156 022626          CMP      (SP)+, (SP)+ ;ADJUST STACK
2618 012160 000745          BR       3$         ;CHECK NEXT BLOCK
2619
2620                                ;NO KT PRESENT - MAP MAX OF 28K IN 4K CONTIGUOUS BLOCKS
2621 012162 012767 000001 166360 MAPMB:  MOV      #1, NOKT    ;SET FLAG TO INDICATE KT11 NOT PRESENT
2622 012170 022626          CMP      (SP)+, (SP)+ ;RESTORE STACK POINTER
2623 012172 012737 012272 000004      MOV      #3$, @#4    ;SET UP TIMEOUT RETURN
2624 012200 012767 000177 166656      MOV      #177, MEML  ;INITIALIZE MAP
2625 012206 005067 166654          CLR      MEMH
2626 012212 012767 000001 166300      MOV      #1, BITPT   ;SETUP 4K POINTER
2627 012220 005001          CLR      R1          ;INITIALIZE BANK ADDRESS
2628 012222 005761 001000 1$:    TST      1000(1)     ;1ST K PRESENT
2629 012226 005761 005000          TST      5000(1)     ;2ND K PRESENT
2630 012232 005761 011000          TST      11000(1)    ;3RD K PRESENT
2631 012236 005761 015000          TST      15000(1)    ;4TH K PRESENT
2632 012242 062701 020000 2$:    ADD      #20000, R1   ;UPDATE TEST ADDRESS
2633 012246 006367 166246          ASL      BITPT       ;UPDATE POINTER TO NEXT 4K
2634 012252 022767 000200 166240      CMP      #200, BITPT ;28K CHECKED YET?
2635 012260 003360          BGT      1$         ;NO, CHECK NEXT 4K BLOCK
2636 012262 012737 000006 000004      MOV      #6, @#4
2637 012270 000207          RTS      %7
2638 012272 046767 166222 166564 3$:    BIC      BITPT, MEML ;TIMEOUT OCCURRED- CLEAR BIT TO
2639                                ;INDICATE 4K BLOCK NOT PRESENT
2640                                ;ADJUST STACK POINTER
2640 012300 022626          CMP      (SP)+, (SP)+
2641 012302 000757          BR       2$
2642
2643
2644
2645
2646                                ;*****
2647                                ;MAP PARITY CORE AND CORRESPONDENCE TO ASSOCIATED REGISTERS
2648                                ;*****
2648 012304 013767 001064 166234 MAPREG: MOV      @MEML, MEMUT ;LOAD MAP OF MEMORY PRESENT IN LOWER 64K
2649 012312 012767 000001 166200      MOV      #1, BITPT   ;INITIALIZE 4K POINTER
2650 012320 012767 000001 166700      MOV      #1, KTSTART ;INDICATE KT11 NOT IN USE
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     ;CLEAR FLAG TO INDICATE FIRST 64K
2656                                ;BEING CHECKED
2657
2658                                ;SET WRITE WRONG PARITY IN ALL REGISTERS PRESENT
2659                                ;THEN WRITE TEST LOCATION VIA DAT0 AND READ TEST LOCATION VIA DAT1
2660                                ;THEN CLEAR WRITE WRONG PARITY IN ALL REGISTERS
2660 012346 012702 000566 MAPRB:  MOV      #MPRD, R2 ;LOAD ADDRESS OF TABLE
2661 012352 032712 000001 1$:    BIT      #1, (2)    ;IS THIS REGISTER PRESENT?
2662 012356 001003          BNE      +10        ;NO, GET NEXT ONE
2663 012360 012772 000004 000000      MOV      #WWP, @ (2) ;YES, SET WRITE WRONG PARITY AND CLEAR REST

```

2664	012366	062702	000010		ADD	#10, R2	
2665	012372	020227	000766		CMP	R2, #TREG	; DONE WITH TABLE?
2666	012376	103765			BLO	1\$; BRANCH IF NOT
2667	012400	016703	166140		MOV	TBANK, R3	; LOAD ADDRESS OF 4K BANK UNDER TEST
2668	012404	066703	166620		ADD	ADRTYP, R3	; ADD ADDRESS OFFSET (EITHER 0,2,4,OR 6)
2669	012410	011313			MOV	(3), (3)	; WRITE WRONG PARITY
2670	012412	005713			TST	(3)	; READ WRONG PARITY
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(2)	; CLEAR WRITE WRONG PARITY IN ALL ; PARITY REGISTERS
2675							
2676	012434	062702	000010		ADD	#10, R2	
2677	012440	020227	000766		CMP	R2, #TREG	
2678	012444	103765			BLO	2\$	
2679	012446	012702	000556	MAPRC:	MOV	#MPRO-10, R2	; INIT FOR ERROR CHECKS
2680	012450	062702	000010	1\$:	ADD	#10, R2	
2681	012452	020227	000766		CMP	R2, #TREG	
2682	012456	002031			BGE	MAPRO	; BRANCH IF DONE WITH TABLE
2683	012464	032712	000001		BIT	#1 (2)	; IS THIS REGISTER PRESENT?
2684	012470	001370			BNE	1\$; NO, GET NEXT ADDRESS
2685	012472	005772	000000		TST	2(R2)	; YES, DID THIS CONTROLLER GET A ; PARITY ERROR?
2686							; NO, CHECK NEXT
2687	012476	100365			BPL	1\$; YES, WHICH 64K IS UNDER TEST?
2688	012500	005767	166046		TST	HIWORD	; BRANCH IF UPPER 64K
2689	012504	001004			BNE	2\$; SET BIT IN MAP FOR THIS PARITY REGISTER
2690	012506	056762	166006	000002	BIS	BITPT, 2(2)	
2691	012514	000403			BR	3\$	
2692	012516	056762	165776	000004	BIS	BITPT, 4(2)	; SET BIT IN MAP FOR THIS PARITY REGISTER
2693	012524	105762	000007	3\$:	TSTB	7(2)	; IS THIS THE FIRST ADDRESS FOUND FOR ; THIS PARITY REGISTER?
2694							; NO, BRANCH
2695	012530	001005			BNE	4\$; YES, RECORD LOW BYTE OF ADDRESS (0,2,4,OR 6)
2696	012532	116762	166472	000006	MOVB	ADRTYP, 6(2)	; INDICATE AN ADDRESS HAS BEEN FOUND FOR ; THIS REGISTER
2697	012540	105262	000007		INCB	7(2)	
2698							
2699	012544	000742		4\$:	BR	1\$; CLEAR BAD PARITY
2700	012546	011313		MAPRO:	MOV	2R3, 2R3	; CHECK FIRST 4 ADDRESSES IN EACH 4K
2701	012550	062767	000032	166452	ADD	#2, ADRTYP	
2702	012556	026727	166446	000010	CMP	ADRTYP, #10	
2703	012564	001402			BEQ	1\$; BRANCH IF FIRST 4 ADDRESSES TESTED
2704	012566	000167	177554		JMP	MAPRB	; IF NOT, GO TEST NEXT ONE
2705	012572	005767	165754	1\$:	TST	HIWORD	; IS LOWER MEMORY DONE?
2706	012576	001021			BNE	MAPRE	; YES, BRANCH
2707	012600	026727	165714	000100	CMP	BITPT, #100	; DONE WITH 1ST 28K?
2708	012606	103015			BHIS	MAPRE	; YES, BRANCH
2709	012610	062767	020000	165726	ADD	#20000, TBANK	; NO- ADD 4K TO ADDRESS
2710	012616	006367	165676		ASL	BITPT	; SHIFT BIT POINTER
2711	012622	005067	166402		CLR	ADRTYP	; START WITH FIRST ADDRESS IN BANK
2712	012626	036767	165666	165712	BIT	BITPT, MEMUT	; DOES THIS 4K BLOCK EXIST?
2713	012634	001756			BEQ	1\$; NO- BRANCH
2714	012636	000167	177504		JMP	MAPRB	; YES, TEST IT
2715	012642	005767	165702	MAPRE:	TST	NOKT	; KTI1 PRESENT?
2716	012646	001401			BEQ	.+4	; YES, BRANCH
2717	012650	000207			RTS	%7	; NO, DONE
2718	012652	005767	166350		TST	KTSTART	; KTI1 ALREADY ON?
2719	012656	001417			BEQ	1\$; YES, BRANCH

2720	012660	012767	020000	165656		MOV	#20000, TBANK	;NO, INIT TBANK TO SELECT KERNEL PAGE 1
2721	012666	012767	001400	165670		MOV	#1400, RELOC	;SET UP FOR ACCESS TO 28K 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								;INDICATE KT11 NOW IN USE
2727	012704	005067	166316			CLR	KTSTART	;TURN ON KT11
2728	012710	012737	000001	177572		MOV	#1, %SR0	;SHIFT BANK INDICATOR
2729	012716	006367	165576		1\$:	ASL	BITPT	;BRANCH IF FIRST 64K NOT DONE
2730	012722	103011				BCC	2\$;IF FIRST 64K DONE, SETUP FOR
2731	012724	012767	000001	165566		MOV	#1, BITPT	;SECOND 64K
2732	012732	013767	001066	165606		MOV	%MEMH, MEMUT	;INDICATE NOW TESTING HIGH 64K
2733	012740	012767	000001	165604		MOV	#1, HIWORD	
2734	012746	062757	000200	165610	2\$:	ADD	#200, RELOC	
2735	012754	022767	007600	165602		CMP	#7600, RELOC	;UP TO EXTERNAL BANK YET?
2736	012762	005003				BGT	3\$;NO, CONTINUE
2737	012764	005037	177572			CLR	%SR0	;YES, TURN OFF KT11 AND EXIT
2738	012770	000207				RTS	%7	
2739	012772	036767	165522	165546	3\$:	BIT	BITPT, MEMUT	;IS THIS 4K PRESENT?
2740	013000	001746				BEQ	1\$;NO- BRANCH
2741	013002	016777	165556	166242		MOV	RELOC, %KPAR1	;YES, MAP PAGE 1 TO THIS BANK
2742	013010	005067	166214			CLR	ADRTP	
2743	013014	000167	177326			JMP	MAPRB	;GO TEST FOR PARITY MEMORY

```

;*****
;ROUTINE TO TYPE MAP OF WHERE PARITY MEMORY IS PRESENT
;AND WHICH CONTROL REGISTERS CONTROL WHICH MEMORY
;*****
TMAP: JSR %7, CLRPAR ;CLEAR ALL PARITY REGISTERS PRESENT
      TYPE ;TYPE "THE PARITY REGISTERS CONTROL MEMORY
      MTMAP ;AS FOLLOWS:"
      MOV #MPRO-10, R1 ;SET UP POINTER
      ADD #10, R1
      CMP R1, %TREG ;DONE WITH MAP?
      BGE TMAPEX ;YES, BRANCH
      CLR HIWORD
      CLR TRFLG ;INITIALIZE TRANSITION FLAG (USED TO
                ;FIGURE MEMORY LIMITS)
      CLR TYFLG ;INITIALIZE TO INDICATE NOTHING TYPED FOR
                ;THIS REGISTER YET
      MOV #-4, TYCOR
      MOV 2(1), ADRPT ;GET LOW 64K MEMORY MAP WORD
      MOV #1, BITPT ;INITIALIZE 4K POINTER
      BIT #1, (1) ;DOES THIS CONTROL EXIST?
      BNE TMAPA ;NO, GET ADDRESS OF NEXT ONE
      MOV (1), TEMPX ;YES, PRINT ITS ADDRESS
      JSR R5, %ACNV
      TEMPX
      MPRAD
      6
      TYPE
      MX1
      TYPE
      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 R5, 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 R5, 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 IS ;NO, BRANCH
2812 013306 005767 165212 TST TYFLG ;YES, WAS ANY PARITY MEMORY
; FOUND FOR THIS REGISTER?
2813
2814
2815 013312 001250 BNE TMAPA
2816 013314 104002 ERROR ;NO PARITY MEMORY WAS FOUND FOR THIS
; REGISTER- EITHER WRITE WRONG PARITY
; FAILED, PARITY ERROR GENERATE OR
; DETECT FAILED, OR THE PARITY ERROR
; BIT FAILED TO SET
2817
2818
2819
2820
2821 013316 000646 BR TMAPA
2822 013320 016167 000004 165170 IS: 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 HIWORD ;INDICATE LOW 64K DONE
2825 013340 000677 BR TMAPB ;LOOP
2826 013342 000207 TMAPPEX: RTS ;RETURN WHEN DONE
2827
2828 ;ROUTINE TO HANDLE BK SYSTEMS
2829 ;
2830 ;
2831 013344 012767 000002 165146 SMLSYS: MOV #2, BITPT ;SET UP POINTER FOR BANK 1

```

```

2832 013352 005037 000042
2833 013356 104000
2834 013360 017466
2835 013362 000207
2836
2837
2838
2839
2840
2841
2842
2843
2844 013364 012767 016517 000266
2845 013372 012767 177777 000262
2846 013400 012767 000240 000256
2847 013406 017767 165354 165124
2848 013414 004567 001776
2849 013420 000766
2850 013422 016526
2851 013424 000006
2852 013426 004567 001764
2853 013432 000540
2854 013434 016550
2855 013436 000006
2856 013440 000461
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866 013442 012767 016517 000210
2867
2868 013450 012767 016561 000204
2869 013456 012767 177777 000200
2870 013464 010567 165042
2871 013470 017767 165272 165042
2872 013476 004567 001714
2873 013502 000766
2874 013504 016526
2875 013506 000006
2876 013510 004567 001702
2877 013514 000540
2878 013516 016550
2879 013520 000006
2880 013522 004567 001670
2881 013526 000532
2882 013530 016605
2883 013532 000006
2884 013534 004567 001656
2885 013540 000534
2886 013542 016621
2887 013544 000006

```

```

CLR 2#42
TYPE
NOMON
RTS PC

```

;MAKE EXIT STANDALONE

```

;*****
;ERROR HANDLER
;*****
;ERRORS CALL ENTERS HERE
;TYPES PC, ICNT, MPR ADDRESS, AND MPR CONTENTS
;TREG SHOULD CONTAIN ADDRESS OF PARITY REGISTER
ERRST.  MOV  #MSTR,ERRB ;SETUP TO TYPE MPR ADDRESS AND CONTENTS
        MOV  #-1,ERRBX
        MOV  #240,ERRBX+2 ;NOP LOCATION AFTER MESSAGE
        MOV  @TREG,TRDATA ;SETUP DATA
        JSR  RS,OACNV ;CONVERT TO ASCII
        TREG
        MTREG
        6
        JSR  RS,OACNV ;CONVERT TO ASCII
        TRDATA
        MDATA
        6
        BR   ERRA

;ERRORP CALL ENTERS HERE
;TYPES PC, ICNT, MPR ADDRESS, MPR CONTENTS
;TEST LOCATION ADDRESS, VALUE EXPECTED, VALUE FOUND
;RS MUST CONTAIN ADDRESS OF TEST LOCATION
;SHDBE MUST CONTAIN EXPECTED VALUE
;WAS MUST CONTAIN ACTUAL DATA
;TREG MUST CONTAIN ADDRESS OF PARITY REGISTER
ERRP:  MOV  #MSTR,ERRB ;IN ADDITION TO BASIC PRINTOUT, TYPE
        MOV  #MSTRX,ERRBX ;MPR ADDRESS AND CONTENTS
        MOV  #-1,ERRBX+2 ;ALSO OUTPUT DATA EXPECTED AND ACTUAL
        MOV  RS,†STLOC ;NOP LOCATION AFTER MESSAGE
        MOV  @TREG,TRDATA ;STORE ADDRESS BEING TESTED
        JSR  RS,OACNV
        TREG
        MTREG
        6
        JSR  RS,OACNV
        TRDATA
        MDATA
        6
        JSR  RS,OACNV
        †STLOC
        MSTRX1
        6
        JSR  RS,OACNV
        SHDBE
        MSTRX3
        6

```

2888 013546 004567 001644
2889 013552 000536
2890 013554 016635
2891 013556 000006
2892 013560 000411
2893
2894
2895
2896
2897
2898 013562 012767 177777 000070
2899 013570 012767 000240 000064
2900 013576 012767 000240 000060
2901 013604 032777 020000 165266
2902 013612 001025
2903 013614 011667 000070
2904 013620 162767 000002 000062
2905 013626 004567 001564
2906 013632 013710
2907 013634 016472
2908 013636 000006
2909 013640 004567 001552
2910 013644 016460
2911 013646 016510
2912 013650 000006
2913 013652 004567 001506
2914 013656 016464
2915 013660 000000
2916 013662 000000
2917 013664 177777
2918 013666 023737 000042 000046
2919 013674 001403
2920 013676 005777 165176
2921 013702 100001
2922 013704 000000
2923 013706 000002
2924 013710 000000
2925
2926
2927
2928
2929 013712 013746 000004
2930 013716 013746 000006
2931 013722 012737 000006 000004
2932 013730 012737 000002 000006
2933 013736 010146
2934 013740 010246
2935 013742 010346
2936 013744 012701 001232
2937 013750 012703 000040
2938 013754 012102
2939 013756 005022
2940 013760 005303
2941 013762 001375
2942 013764 020127 001236
2943 013770 003767

JSR R5,OACNV
WAS
MSTRXS
6
BR ERRA
;ERROR CALL ENTERS HERE
;TYPE PC AND ICNT ONLY
ERR: MOV #-1,ERRB ;SET UP ONE MESSAGE CALL
MOV #240,ERRBX
MOV #240,ERRBX+2
ERRA: BIT #BIT13,JSWR ;INHIBIT ERROR PRINT?
BNE ERRC ;YES- BRANCH
MOV (SP),ERRD ;NO- DEVELOP CALLING ADDRESS
SUB #2,ERRD
JSR R5,OACNV ;GO TO OCTAL TO ASCII CONVERT
ERRA1: ERRC ;SOURCE ADDRESS
ERRA2: MPC ;DESTINATION ADDRESS
6 ;#OF DIGITS TO CONVERT
JSR R5,OACNV ;CONVERT ICNT TO ASCII
ERRA3: ICNT
ERRA4: MICNT
6
JSR R5,TYPSX ;TYPE MESSAGE
ERRA5: MEO ;ERROR HEADER
ERRB: OPEN ;ADDITIONAL ERROR MESSAGES IF ANY
ERRBX: OPEN
-1
ERRC: CMP @#42,@#46 ;ARE WE IN ACT11 AUTOMATIC MODE?
BEQ +10 ;YES HALT ON ERROR
TST @SWR ;HALT ON ERROR SET?
BPL +4 ;NO- BRANCH
HALT ;YES- ERROR OCCURRED SO HALT
RTI
ERRD: OPEN
;MAP ALL PAGES NON-RESIDENT, BANK 0
NRALL: MOV @#4,-(SP)
MOV @#6,-(SP)
MOV #6,@#4
MOV #RTI,@#6
MOV R1,-(SP)
MOV R2,-(SP)
MOV R3,-(SP)
MOV #PORTAB,R1
1\$: MOV #32,R3
MOV (R1)+,R2
2\$: CLR (R2)+
DEC R3
BNE 2\$
CMP R1,#PDREND
BLE 1\$

;SET UP ONE MESSAGE CALL
;INHIBIT ERROR PRINT?
;YES- BRANCH
;NO- DEVELOP CALLING ADDRESS
;GO TO OCTAL TO ASCII CONVERT
;SOURCE ADDRESS
;DESTINATION ADDRESS
;#OF DIGITS TO CONVERT
;CONVERT ICNT TO ASCII
;TYPE MESSAGE
;ERROR HEADER
;ADDITIONAL ERROR MESSAGES IF ANY
;ARE WE IN ACT11 AUTOMATIC MODE?
;YES HALT ON ERROR
;HALT ON ERROR SET?
;NO- BRANCH
;YES- ERROR OCCURRED SO HALT

2944	013772	012603	
2945	013774	012602	
2946	013776	012601	
2947	014000	012637	000006
2948	014004	012637	000004
2949	014010	000207	

```

MOV (SP)+,R3
MOV (SP)+,R2
MOV (SP)+,R1
MOV (SP)+,R#6
MOV (SP)+,R#4
RTS %7

```

2950			
2951			
2952			
2953			

:ROUTINE TO CLEAR ALL PARITY REGISTERS PRESENT

2954	014012	010146	
2955	014014	010246	
2956	014016	010701	
2957	014020	062701	164546
2958	014024	010702	
2959	014026	062702	164740
2960	014032	032711	000001
2961	014036	001002	
2962	014040	005071	000000
2963	014044	062701	000010
2964	014050	020102	
2965	014052	103767	
2966	014054	012602	
2967	014056	012601	
2968	014060	000207	

```

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

```

2969			
2970			
2971			
2972			

:ROUTINE TO MAP KERNEL 0 TO BANK 0 READ/WRITE.
:KERNEL 1 READ/WRITE BUT BANK MAPPED BY CALLING ROUTINE.
:AND KERNEL 7 TO EXTERNAL BANK, READ/WRITE

2973			
2974			
2975	014062	012777	077406 165150
2976	014070	012777	077406 165144
2977	014076	012777	077406 165142
2978	014104	012777	007600 165144
2979	014112	005077	165132
2980	014116	012737	000001 177572
2981	014124	000207	

```

MAP1: MOV #77406, RKPDR0
MOV #77406, RKPDR1
MOV #77406, RKPDR7
MOV #7600, RKPDR7
CLR RKPDR0
MOV #1, RSRD
RTS %7

```

2982			
2983			
2984			
2985			

:ROUTINE TO LOCATE THE FIRST PARITY MEMORY ADDRESS (ABOVE BANK 0)
:CORRESPONDING TO A GIVEN PARITY REGISTER-REQUIRES THAT THE ROUTINES
:MAPMEM AND MAPREG HAVE ALREADY BEEN RUN
:TO USE, PUT THE ADDRESS OF THE ADDRESS OF THE REGISTER IN R0 (I.E. POINT TO
:TO MAP TABLE)-THE DESIRED ADDRESS IS RETURNED IN R1, USING KERNEL PAGE 1 IF
:KT11 IS PRESENT

2990			
2991	014126	010246	
2992	014130	012702	000200
2993	014134	012701	000002
2994	014140	005767	164404
2995	014144	001403	
2996	014146	022701	000200
2997	014152	101420	
2998			
2999	014154	030160	000002

```

LOCATM: MOV R2, -(SP)
MOV #200, R2
MOV #2, R1 ;SKIP USE OF BANK 0
1$: TST NOKT ;KT11 PRESENT?
BEQ 3$ ;YES, BRANCH
CMP #200, R1 ;NO, CHECK ONLY FOR MEMORY IN FIRST 28K
BLOS 4$ ;IF NO MEMORY FOUND IN 1ST 28K, GIVE
;ERROR RETURN
3$: BIT R1, 2(R0) ;DOES THIS 4K CORRESPOND TO THIS REGISTER?

```

```

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$:    MOV     #1,R1           ;NO PARITY MEMORY CORRESPONDS TO
3011 014220 012602          MOV     (SP)+,R2        ;THIS REGISTER- RETURN WITH ERROR
3012 014222 000207          RTS                     ;INDICATION
3013 014224 005767 164320  LOCAT1: TST    NOKT     ;KT11 PRESENT?
3014 014230 001005          BNE     1$             ;NO- BRANCH OVER
3015 014232 010277 165014  MOV     R2,2KPAR1      ;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     ;
3026 014270 066701 000004  ADD     LSAV,R1         ;
3027 014274 012602          MOV     (SP)+,R2        ;RESTORE R2
3028 014276 000207          RTS                     ;AND RETURN
3029 014300 000000          LSAV:   0              ;
3030
3031
3032
3033          ;SAVE LOADER IN TOP OF FIRST 4K
3034 014302 013746 000004  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
3038 014326 010146          MOV     R1,-(SP)       ;SETUP TO RTI ON TIMEOUT
3039 014330 010246          MOV     R2,-(SP)       ;SAVE REGISTERS
3040 014332 012701 152234          MOV     #152234,R1
3041 014336 000261          1$:    SEC
3042 014340 005711          TST     @R1
3043 014342 103006          BCC     2$             ;IF TIMEOUT, C BIT WILL STILL BE SET
3044 014344 162701 020000          SUB     #20000,R1      ;IF NO TIMEOUT, C BIT WILL BE CLEAR
3045 014350 020127 020000          CMP     R1,#20000      ;TIMEOUT OCCURRED, CHECK FOR NEXT LOWER BANK
3046 014354 101370          BHI     1$             ;
3047 014356 000410          BR      SAVLDX         ;ONLY 4K OF 1ST 28K PRESENT- EXIT
3048 014360 012702 032234 2$:    MOV     #32234,R2      ;THIS BANK IS HIGHEST OF 1ST 28K- COPY
3049 014364 012122 040000 3$:    MOV     (R1)+(R2)+    ;LOADER AREA TO BANK 0
3050 014366 020227          CMP     R2,#40000
3051 014372 103774          BLO     3$             ;
3052 014374 005267 000016          INC     LDRSVD         ;INDICATE LOADER HAS BEEN MOVED TO
3053
3054          SAVLDX: MOV     (SP)+,R2    ;THE TOP OF THE FIRST 4K
3055 014402 012601          MOV     (SP)+,R1

```


3056 014404 012637 000006
 3057 014410 012637 000004
 3058 014414 000207
 3059 014416 000000

MOV (SP)+,2#6
 MOV (SP)+,2#4
 RTS %7
 LDRSVD: 0

3060
 3061
 3062
 3063
 3064

;ROUTINE TO RESTORE THE LOADER FROM BANK 0 (WHERE IT WAS SAVED) TO THE
 ;HIGHEST BANK IN THE FIRST 28K OF MEMORY

3065 014420 005767 177772
 3066 014424 001431
 3067 014426 012737 000006
 3068 014434 012737 000002
 3069 014442 012701 152234

RSTLDR: TST LDRSVD
 BEQ RSTLDX
 MOV #6,2#4
 MOV #R1,2#6
 MOV #152234,R1

000004
 000006

3070 014446 000261
 3071 014450 005711
 3072 014452 103006
 3073 014454 162701 020000
 3074 014460 020127 020000

1\$: SEC
 TST 2R1
 BCC 2\$
 SUB #20000,R1
 CMP R1,#20000

; IF TIMEOUT, C BIT WILL BE SET
 ; IF NO TIMEOUT, C BIT WILL BE CLEAR
 ; TIMEOUT OCCURRED, CHECK FOR NEXT
 ; LOWER BANK

3075 014464 101370
 3076 014466 000410
 3077 014470 012702 032234
 3078 014474 012221
 3079 014476 020227 040000

2\$: BR RSTLDX
 3\$: MOV #32234,R2
 MOV (R2)+(R1)+
 CMP R2,#40000
 BLO 3\$

;TYPE MESSAGE "LOADER RESTORED"

3080 014502 103774
 3081 014504 104000
 3082 014506 017506
 3083 014510 000207
 3084 014512 000776

RSTLDR: LDRMSG
 RTS PC
 BR -2

;LOADER HAS BEEN RESTORED
 ;TO HIGHEST BANK IN FIRST 28K

3085
 3086
 3087
 3088
 3089

;SCAN ALL MEMORY FOR BAD PARITY, TYPE 18 BIT ADDRESSES OF
 ;LOCATIONS FOUND TO BE BAD, AND WRITE INTO LOCATIONS WITH GOOD PARITY
 PSCAN: MOV R1,-(SP) ;STORE REGISTERS AND LOCATIONS TO BE
 MOV R2,-(SP) ;ALTERED
 MOV R3,-(SP)
 MOV R4,-(SP)
 MOV 2#4,-(SP)
 MOV 2#6,-(SP)
 MOV 2#114,-(SP)
 MOV 2#116,-(SP)

3090 014514 010146
 3091 014516 010246
 3092 014520 010346
 3093 014522 010446
 3094 014524 013746 000004
 3095 014530 013746 000006
 3096 014534 013746 000114
 3097 014540 013746 000116
 3098 014544 012737 000006
 3099 014552 005037 000006
 3100 014556 012737 000116
 3101 014564 005037 000116
 3102 014570 005767 163754
 3103 014574 001513
 3104 014576 005002
 3105 014600 012703 000001
 3106 014604 004767 177202
 3107 014610 005712
 3108 014612 000240
 3109 014614 012701 000566
 3110
 3111 014620 032711 000001

000004

000114

;SETUP TIMEOUT TRAPCATCHER

;SETUP PARITY TRAP TRAPCATCHER

;KT11 PRESENT?

;YES, BRANCH

;R2 CONTAINS TEST ADDRESS

;R3 USED AS A BIT POINTER

;CLEAR ALL PARITY REGISTERS

;READ LOCATION TO CHECK FOR BAD PARITY

;SETUP TO SCAN REGISTERS FOR PARITY

;ERROR SET

3109 014614 012701 000566
 3110
 3111 014620 032711 000001

1\$: JSR %7,CLRPAR
 TST 2R2
 NOP
 MOV #MPRO,R1

2\$: MOV #1,R3
 3\$: BIT #1,2R1

3112	014624	001003		BNE	.+10		
3113	014626	005771	000000	TST	2(R1)		: PARITY ERROR SET?
3114	014632	100424		BMI	6\$: YES- BRANCH
3115	014634	062701	000010	ADD	#10, R1		: NO- CHECK NEXT REGISTER
3116	014640	020127	000766	CMP	R1, #TREG		
3117	014644	103765		BLO	3\$: LOOP UNTIL ALL REGISTERS HAVE BEEN CHECKED
3118	014646	062702	000002	4\$: ADD	#2, R2		: MOVE ADDRESS POINTER
3119	014652	032702	017777	BIT	#17777, R2		: DONE WITH 4K?
3120	014656	001352		BNE	1\$: NO- CONTINUE
3121	014660	006303		5\$: ASL	R3		: YES, CHECK FOR TESTING NEXT 4K
3122	014662	020327	000200	CMP	R3, #200		
3123	014666	103035		BHS	PSCANX		: EXIT IF DONE WITH 28K
3124	014670	030367	164170	BIT	R3, MEML		: IS THIS MEMORY PRESENT?
3125	014674	001343		BNE	1\$: YES, GO TEST IT
3126	014676	062702	020000	ADD	#20000, R2		: NO- UPDATE ADDRESS
3127	014702	000766		BR	5\$: LOOP
3128	014704	010267	000110	6\$: MOV	R2, PSADRS		: PARITY ERROR OCCURRED
3129	014710	004567	000502	JSR	R5, OACNV		: GET ASCII OF ADDRESS CONTAINING BAD PARITY
3130	014714	015020		PSADRS			
3131	014716	017440		MPSER1			
3132	014720	000006		6			
3133	014722	104000		TYPE			: TYPE MESSAGE "BAD PARITY FOUND IN LOCATION"
3134	014724	017410		MPSER			: AND ADDRESS OF FAILING LOCATION
3135	014726	032777	002000	7\$: BIT	#2000, 2SWR		: SWITCH 10 SET?
3136	014734	001401		BEQ	.+4		: NO- CONTINUE
3137	014736	000000		HALT			: HALT ON BAD PARITY SET
3138	014740	011212		MOV	2R2, 2R2		: WRITE INTO LOCATION- SHOULD
3139							: CLEAR BAD PARITY
3140	014742	005071	000000	CLR	2(R1)		: CLEAR CORRESPONDING PARITY REGISTER
3141	014746	005712		TST	2R2		: READ LOCATION TO SEE IF BAD PARITY WAS
3142	014750	005771	000000	TST	2(R1)		: CLEARED
3143	014754	100001		BPL	.+4		: OK- BRANCH
3144	014756	104002		ERROR			: BAD PARITY DIDN'T CLEAR WHEN LOCATION
3145							: WAS REWRITTEN
3146	014760	000732		BR	4\$: GO CHECK NEXT LOCATION
3147	014762	004767	177024	8\$: JSR	PC, CLRPAR		: DONE- CLEAR ALL PARITY REGISTERS
3148	014766	012637	000115	MOV	(SP)+, 2#116		: RESTORE LOCATIONS ALTERED
3149	014772	012637	000114	MOV	(SP)+, 2#114		
3150	014776	012637	000006	MOV	(SP)+, 2#6		
3151	015002	012637	000004	MOV	(SP)+, 2#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		: RETURN
3157	015020	000000		PSADRS:	0		
3158	015022	000000		PSCANH:	0		
3159							
3160							
3161							
3162							
3163							
3164	015024	017746	164222	9\$: MOV	2KPAR1, -(SP)		: SCAN ALL MEMORY FOR BAD PARITY USING KT11
3165	015030	032737	000001	177572	BIT	#1, 2#SR0	: TYPE 18 BIT ADDRESSES OF LOCATIONS FOUND BAD, AND WRITE GOOD PARITY BACK IN
3166	015036	001004		BNE	1\$: SAVE CONTENTS OF KERNEL PAR1
3167	015040	004767	176646	JSR	PC, NRALL		: SKIP IF KT11 IS ALREADY ON

```

3168 015044 004767 177012      JSR      PC,MAP1
3169
3170 015050 005067 177746      1$:     CLR      PSCANH
3171 015054 005077 164172      CLR      @KPAR1
3172 015060 012703 000001      PSL00P: MOV     #1,R3
3173 015064 005767 177732      PSLUP:  TST     PSCANH
3174 015070 001004          BNE     2$
3175 015072 030367 163766      BIT     R3,MEML
3176 015076 001022          BNE     PSXTST
3177 015100 000403          BR      PSNXT
3178 015102 030367 163760      2$:     BIT     R3,MEMH
3179 015106 001016          BNE     PSXTST
3180 015110 062777 000200 164134 PSNXT:  ADD     #200,@KPAR1
3181 015116 006303          ASL     R3
3182 015120 103361          BCC     PSLUP
3183 015122 005767 177674      TST     PSCANH
3184 015126 001003          BNE     PSCX1
3185 015130 005267 177666      INC     PSCANH
3186
3187 015134 000751          BR      PSL00P
3188 015136 012677 164110      PSCX1: (SP)+,@KPAR1
3189 015142 000707          BR      PSCANX
3190 015144 012702 020000      PSXTST: MOV     #20000,R2
3191 015150 004767 176636      1$:     JSR     @R2
3192 015154 005712          TST     @R2
3193 015156 012701 000566          MOV     #MPRU,R1
3194 015162 032711 000001      2$:     BIT     #1,R1
3195 015166 001003          BNE     +10
3196 015170 005771 000000          TST     @R1
3197 015174 100413          BMI     4$
3198 015176 062701 000010          ADD     #10,R1
3199 015202 020127 000766          CMP     R1,#TREG
3200 015206 103765          BLO     2$
3201 015210 062702 000002      3$:     ADD     #2,R2
3202 015214 020227 040000          CMP     R2,#40000
3203 015220 103753          BLO     1$
3204 015222 000732          BR      PSNXT
3205 015224 010267 177570      4$:     MOV     R2,PSADRS
3206 015230 042767 160000 177562      BIC     #16000,PSADRS
3207 015236 005046          CLR     -(SP)
3208 015240 017746 164006          MOV     @KPAR1,-(SP)
3209 015244 006316          ASL     @SP
3210 015246 006316          ASL     @SP
3211 015250 006316          ASL     @SP
3212 015252 006316          ASL     @SP
3213 015254 006316          ASL     @SP
3214 015256 006166          ROL     2(SP)
3215 015262 006316          ASL     @SP
3216 015264 006166          ROL     2(SP)
3217 015270 006366          ASL     2(SP)
3218 015274 062667 177520          ADD     (SP)+,PSADRS
3219 015300 004567 000112      JSR     R5,OACNV
3220 015304 015020          PSADRS
3221 015306 017440          MPSER1
3222 015310 000006          6
3223 015312 116704 002122      MOVB   MPSER1,R4

```

```

;MAP KERNEL 7 TO THE EXTERNAL BANK, RW
;MAP KERNEL 1 RW, AND TURN ON KT11
;CLEAR FLAG TO INDICATE CHECKING FIRST 64K
;INITIALIZE TO BANK 0
;R3 IS USED AS A BIT POINTER
;TESTING TOP 64K?
;YES, BRANCH
;NO- IS PARITY MEMORY PRESENT IN THIS 4K?
;YES- GO TEST IT
;NO- CHECK FOR NEXT 4K
;IS PARITY MEMORY PRESENT IN THIS 4K?
;YES- GO TEST IT
;NO- MAP TO NEXT 4K

;BRANCH IF NOT END OF 64K
;END OF TOP 64K?
;YES, GET READY TO EXIT
;NO, SET FLAG INDICATING DONE WITH
;LOWER 64K

```

```

;R2 USED AS ADDRESS POINTER
;CLEAR ALL PARITY REGISTERS
;READ LOCATION
;SETUP TO SCAN REGISTERS FOR PARITY ERROR SET

```

```

;PARITY ERROR SET?
;YES, BRANCH
;NO, CHECK NEXT

```

```

;LOOP UNTIL ALL REGISTERS HAVE BEEN CHECKED
;UPDATE TEST ADDRESS POINTER
;DONE WITH BANK?
;NO- LOOP
;YES- GO CHECK FOR ANOTHER BANK
;PARITY ERROR OCCURRED- GET 18 BIT
;OCTAL ADDRESS OF BAD LOCATION

```

```

;CONVERT LOW 16 OCTAL BITS TO ASCII

```

3224	015316	062604			ADD	(SP)+,R4	;CHANGE TO ASCII FOR 18 BITS
3225	015320	110467	002114		MOVB	R4,MPSER1	
3226	015324	104000			TYPE		;TYPE ADDRESS OF LOCATION WITH BAD PARITY
3227	015326	017410			MPSER		
3228	015330	032777	002000	163542	BIT	#2000,@SWR	;SWITCH 10 SET?
3229	015336	001401			BEQ	.+4	;NO- BRANCH
3230	015340	000000			HALT		;HALT ON BAD PARITY SET
3231	015342	011212			MOV	@R2,@R2	;REWRITE LOCATION CONTAINING BAD PARITY
3232	015344	005071	000000		CLR	@(R1)	;CLEAR PARITY ERROR BIT
3233	015350	005712			TST	@R2	;READ LOCATION TO SEE IF PARITY IS NOW GOOD
3234	015352	005771	000000		TST	@(R1)	;CHECK PARITY ERROR BIT
3235	015356	103001			BPL	.+4	
3236	015360	104002			ERROR		;REWRITING LOCATION DID NOT CLEAR BAD PARITY
3237	015362	000712			BR	35	;GO TEST NEXT LOCATION
3238							
3239							
3240							
3241							
3242							
3243	015364	012567	000022		;PIC ROUTINE TO OUTPUT A SERIES OF ASCII MESSAGES (CALLED VIA JSR R5)		
3244	015370	022767	177777	000014	TYP5X:	MOV (R5)+,TYP5BX	;GET ADDRESS OF MESSAGE
3245	015376	001001				CMPI,TYP5BX	;TERMINATOR?
3246	015400	000205				BNE TYP5AX	;NO BRANCH
3247	015402	013746	177776		TYP5AX:	RTS	;YES, RETURN
3248	015406	004767	163502			MOV @#PS,-(SP)	;SETUP TO CALL TYPE ROUTINE VIA JSR
3249	015412	000000			JSR	PC,\$TYPE	;TYPE ASCII MESSAGE
3250	015414	000763			TYP5BX:	OPEN	
3251					BR	TYP5X	
3252							
3253							
3254							
3255	015416	013567	000074		;SUBROUTINE FOR OCTAL TO ASCII CONVERSION		
3256	015422	012567	000072		OACNV:	MOV @5+,OACNVX	;GET OCTAL VALUE
3257	015426	012567	000070			MOV (5)+,OACDST	;GET DESTINATION ADDRESS
3258	015432	066767	000064	000060		MOV (5)+,OACNT	;GET CONVERT COUNT
3259	015440	016746	000052		OACNVA:	ADD OACNT,OACDST	;DEVELOP ADDRESS TO STORE 1ST CHAR.
3260	015444	042716	177770			MOV OACNVX,-(SP)	
3261	015450	062716	000060			BIC #177770,@SP	;ISOLATE LEAST SIGNIFICANT DIGIT
3262	015454	005367	000040			ADD #60,@SP	;CONVERT DIGIT TO ASCII
3263	015460	112677	000034			DEC OACDST	
3264	015464	042767	000007	000024		MOVB (SP)+,@OACDST	;STORE ASCII CHARACTER
3265	015472	006067	000020			BIC #7,OACNVX	
3266	015476	006067	000014			ROR OACNVX	
3267	015502	006067	000010			ROR OACNVX	
3268	015506	005367	000010			ROR OACNVX	
3269	015512	001352				DEC OACNT	;DONE ALL DIGITS?
3270	015514	000205				BNE OACNVA	;BRANCH IF NOT DONE
3271	015516	000000				RTS	;DONE, EXIT
3272	015520	000000			OACNVX:	OPEN	
3273	015522	000000			OACDST:	0	
3274					OACNT:	0	
3275							
3276							
3277							
3278	015524	104005			;SUBROUTINE FOR BINARY TO DECIMAL ASCII CONVERSION		
3279	015526	012700	015702		BDCNV:	SAV04	;SAVE REGS
						MOV #DECVAL,%0	;SET UP ADDR TO STORE DECIMAL ASCII

```

3280 015532 013501          MOV      2(5)+,R1          ;BINARY VALUE TO R1
3281 015534 012567 000052    MOV      (5)+,BDCNVC      ;DESTINATION ADDR TO BDCNVC
3282 015540 012567 000050    MOV      (5)+,BDCNVD      ;CHARACTER COUNT TO BDCNVD
3283 015544 012702 015670          MOV      #ADTENP,R2      ;ADDR OF TEN POWER STRING
3284 015550 012767 000005 000104    MOV      #5,CNVCTR        ;SET UP FOR 5 POWER CONVERSIONS
3285 015556 012267 000104    BDCNVA: MOV      (2)+,TENPWR    ;MOVE POWER OF TEN VALUE
3286 015562 004767 000034          JSR      PC,SUBTEN        ;PERFORM CONVERSION
3287 015566 005367 000070          DEC      CNVCTR           ;DONE 5 CONVERSIONS?
3288 015572 001371          BNE      BDCNVA          ;BRANCH IF NOT YET 5.
3289 015574 166700 000014          SUB      BDCNVD,%0
3290 015600 010067 000004          MOV      %0,BDCNVB
3291 015604 004567 000100          JSR      R5,BMOVE
3292 015610 000700          BDCNVB: OPEN
3293 015612 000000          BDCNVC: OPEN
3294 015614 000000          BDCNVD: OPEN
3295 015616 104006          RSTO4          ;RESTORE REGS AND EXIT
3296 015620 000205          RTS
3297 015622 005067 000036    SUBTEN: CLR      R5
3298 015626 166701 000034    SUBTNA: SUB      DIGIT
3299 015632 103403          BCS      TENPWR,R1      ;SUBTRACT TEN POWER FROM BINARY VALUE
3300 015634 005267 000024          BCS      SUBTNB          ;BRANCH IF UNSUCCESSFUL SUBTRACTION
3301 015640 000772          BR
3302 015642 066701 000020    SUBTNB: ADD      DIGIT
3303 015646 062767 000060 000010    ADD      SUBTNB          ;RESTORE SUBTRACTED VALUE.
3304 015654 116720 000004          MOV      #60,DIGIT      ;CONVERT (DIGIT) TO ASCII
3305 015660 000207          RTS                ;MOVE ASCII CHAR TO DECVAL FIELD
3306 015662 000000          CNVCTR: OPEN
3307 015664 000000          DIGIT: OPEN
3308 015666 000000          TENPWR: OPEN
3309 015670 023420          ADTENP: 10000.
3310 015672 001750          1000.
3311 015674 000144          100.
3312 015676 000012          10.
3313 015700 000001          1.
3314 015702          040          040          040          DECVAL: .BYTE 040,040,040,040,040,040
3315 015705          040          040
3316
3317
3318
3319          ;SUBROUTINE TO MOVE A VARIABLE NUMBER OF BYTES
3320 015710 104005    BMOVE: SAVO4          ;SAVE REGS
3321 015712 012501          MOV      (5)+,R1          ;GET FROM ADDRESS
3322 015714 012502          MOV      (5)+,R2          ;GET TO ADDRESS
3323 015716 012503          MOV      (5)+,R3          ;GET COUNT
3324 015720 112122    BMOVA: MOV      (1)+,(2)+    ;MOVE BYTE
3325 015722 005303          DEC      R3              ;DECREMENT COUNT
3326 015724 001375          BNE      BMOVA           ;BRANCH IF NOT DONE
3327 015726 104006          RSTO4          ;RESTORE REGS AND EXIT
3328 015730          RTS      R5
3329
3330
3331
3332          ;UNEXPECTED POWER FAIL SERVICE
3333          ;BECAUSE WWP MAY BE SET IN MPR'S AND ALL PROCESSOR REGISTERS
3334          ;MAY BE IN USE, CONTINUATION AFTER POWER FAIL IS NOT ATTEMPTED.
3335          ;INSTEAD, THE PROGRAM RESTARTS AFTER A POWER FAILURE

```

3336	015732	012737	015776
3337	015740	012701	000566
3338	015744	032711	000001
3339	015750	001002	
3340	015752	005071	000000
3341	015756	062701	000010
3342	015762	020127	000766

```

000024 PWRDN: MOV    #PWRUP, @#24
          MOV    #MPRO, R1
          IS:   BIT    #1, @R1
          BNE   .+6
          CLR   @R1
          ADD   #10, R1
          CMP   R1, #TREG

```

;SET UP FOR POWER UP

;CLEAR PARITY REGISTERS IN CASE
;WWP IS SET

```

3343 015766 103766          BLO      1$
3344 015770 010667 163264  MOV     SP,SPSAV
3345 015774 000000          HALT
000024 PWRUP: 015732 000024  MOV     #PWRDN,2#24 ;POWER DOWN HALT
3346 015776 012737 015732  MOV     SPSAV,SP ;SET UP FOR POWER DOWN
3347 016004 016706 163250  CLR     #0 ;STALL SO OUTPUT WON'T BE GARBLED
3348 016010 005027 000000  DEC     -2
3349 016014 005367 177772  BNE     -4 ;TYPE RECOVERY MESSAGE
3350 016020 001375
3351 016022 104000          TYPE
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     2(SP),(SP) ;GET CALL
3361 016044 121667 000050  CMPB   (SP),EMTLIM ;CHECK IF CALL WITHIN LIMITS
3362 016050 101402          BLOS   EMTA ;CALL IS NOT WITHIN LIMITS
3363 016052 000000          HALT
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     2(SP),2(SP)
3369 016074 000136          JMP     2(SP)+ ;GO TO EMT RETURN
3370
3371          :EMT DEFINITIONS AND ASSIGNMENTS
3372          EMTTAB:
3373 016076          TYPE=EMT+EMTX
3374          $TYPE
3375 016076 001114          SCOPE=EMT+EMTX
3376          SCOPEC
3377 016100 016356          ERROR=EMT+EMTX
3378          ERR
3379 016102 013562          ERRORP=EMT+EMTX
3380          ERRP
3381 016104 013442          ERRORS=EMT+EMTX
3382          ERRST
3383 016106 013364          SAV04=EMT+EMTX
3384          SV04
3385 016110 016122          RST04=EMT+EMTX
3386          RSD4
3387 016112 016210          RST05S=EMT+EMTX
3388          RS05S
3389 016114 016236          SAV05S=EMT+EMTX
3390          SV05S
3391 016116 016142          EMTLIM: EMTX-1
3392 016120 000010
3393
3394          :SUBROUTINE TO SAVE REGS 0-4
3395          SV04: MOV     (SP)+,-12.(SP) ;MOVE PC+PS UP STACK
3396 016122 012666 177764  MOV     (SP)+,-12.(SP)
3397 016126 012666 177764  MOV     #RTI,SV05C
3398 016132 012767 000002 000040

```

```

3399 016140 000411 BR SV058
3400
3401
3402
3403 ; SUBROUTINE TO SAVE REGS 0-5 + PLACE EMT PC IN R5
3404 016142 012747 000240 000030 SV05S: MOV #NOP,SV05C
3405 016150 000400 BR SV05A
3406 ; SUBROUTINE TO SAVE REGS 0-5
3407
3408 016152 012666 177762 SV05A: MOV (SP)+,-14.(SP)
3409 016156 012666 177762 MOV (SP)+,-14.(SP)
3410 016162 010546 MOV R5,-(SP)
3411 016164 010446 SV05B: MOV R4,-(SP)
3412 016166 010346 MOV R3,-(SP)
3413 016170 010246 MOV R2,-(SP)
3414 016172 010146 MOV R1,-(SP)
3415 016174 010046 MOV %0,-(SP)
3416 016176 024646 SV05C: CMP -(SP),-(SP)
3417 016200 000002 ; RTI OR NOP
3418 016202 016605 000020 MOV 16.(SP),R5 ; EMT PC TO R5
3419 016206 000002 RTI
3420
3421
3422 ; SUBROUTINE TO RESTORE REGS 0-4
3423
3424 016210 022626 R504: CMP (SP)+,(SP)+
3425 016212 012600 MOV (SP)+,%0
3426 016214 012601 MOV (SP)+,R1
3427 016216 012602 MOV (SP)+,R2
3428 016220 012603 MOV (SP)+,R3
3429 016222 012604 MOV (SP)+,R4
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
3437 016236 010566 000020 R505S: MOV R5,16.(SP) ; SET EMT PC TO R5
3438 016242 022626 CMP (SP)+,(SP)+
3439 016244 012600 MOV (SP)+,%0
3440 016246 012601 MOV (SP)+,R1
3441 016250 012602 MOV (SP)+,R2
3442 016252 012603 MOV (SP)+,R3
3443 016254 012604 MOV (SP)+,R4
3444 016256 012605 MOV (SP)+,R5
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 016272 005037 177776 TFSTX: CLR @#PS
3458 016276 000000 HALT
3459 016300 017767 162574 000154 MOV @SWR, RETURN
3460 016306 062767 000002 000146 ADD #2, RETURN
3461 016314 000000 HALT
3462 016316 012767 177777 162164 MOV #-1 TSTX
3463 016324 032777 010000 162546 BIT #10000, @SWR
3464 016332 001404 BEQ .+12
3465 016334 042737 000020 177776 BIC #20 @#PS
3466 016342 000403 BR .+10
3467 016344 052737 000020 177776 BIS #20 @#PS
3468 016352 000177 000104 JMP @RETURN

```

```

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

```

```

3470
3471
3472 ;SCOPE AND/OR ITERATION LOOP FOR EACH TEST 64 TIMES
3473 ;A SETUP ROUTINE SHOULD INITIALIZE RETURN AND IMAX
3474 016356 032777 040000 162514 SCOPEC: BIT #40000, @SWR
3475 016364 001020 BNE SCOPEB
3476 016366 032777 004000 162504 BIT #4000 @SWR
3477 016374 001021 BNE SCOPEB
3478 016376 005767 173350 TST PASCNT
3479 016402 001416 BEQ SCOPEG
3480 016404 005767 162100 TST TSTX
3481 016410 001006 BNE SCOPEB
3482 016412 026767 000042 000036 CMP ICNT, IMAX
3483 016420 100007 BPL SCOPEG
3484 016422 005267 000032 INC ICNT
3485 016426 022606 SCOPEB: CMP (6)+, %6
3486 016430 012677 161342 MOV (6)+, @PS
3487 016434 000177 000022 JMP @RETURN
3488 016440 005067 162044 SCOPEG: CLR TSTX
3489 016444 005067 000010 CLR ICNT
3490 016450 011667 000006 MOV @%6, RETURN
3491 016454 000002 RTI
3492 016456 000100 IMAX: 100
3493 016460 000000 ICNT: 0
3494 016462 000000 RETURN: 0

```

```

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

```

```

3495
3496
3497
3498 ;ASCII MESSAGES
3499
3500 016464 MED:
3501 016464 005015 041520 020075 MTNUM: .ASCII <15><12>'PC='
3502 016472 020040 020040 020040 MPC: .ASCII ' ICNT='
3503 016500 020040 041511 052116
3504 016506 020075
3505 016510 020040 020040 020040 MICNT: .ASCII ' '
3506 016516 000 MSTR: .ASCII ' MPR='
3507 016517 040 046440 051120
3508 016524 020075
3509 016526 020040 020040 020040 MTREG: .ASCII ' MPR DATA='
3510 016534 020040 050115 020122

```

3511	016542	040504	040524	020075				
3512	016550	020040	020040	020040	MDATA:	.ASCIZ	'	'
3513	016556	020040	000					
3514	016561	015	020012	020040	MSTRX:	.ASCII	<15><12>'	TEST LOC= '
3515	016566	020040	020040	052040				
3516	016574	051505	020124	047514				
3517	016602	036503	040					
3518	016605	040	020040	020040	MSTRX1:	.ASCII	'	'
3519	016612	040						
3520	016613	040	027523	035102		.ASCII	'	S/B: '
3521	016620	040						
3522	016621	040	020040	020040	MSTRX3:	.ASCII	'	'
3523	016626	040						
3524	016627	040	040527	035123		.ASCII	'	WAS: '
3525	016634	040						
3526	016635	040	020040	020040	MSTRX5:	.ASCIZ	'	'
3527	016642	020040	000					
3528	016645	015	051412	052105	MSETSR:	.ASCIZ	<15><12>'	SET SR OPTIONS'
3529	016652	051440	020122	050117				
3530	016660	044524	047117	000123				
3531	016666	020054	051120	051505	MCON:	.ASCIZ	'	, PRESS CONTINUE'
3532	016674	020123	047503	052116				
3533	016702	052516	000105					
3534	016706	005015	042523	020124	MMDEV:	.ASCIZ	<15><12>'	SET DEVICE ADDRESS IN SR'
3535	016714	042504	044523	042503				
3536	016722	040440	042104	042522				
3537	016730	051523	044440	020116				
3538	016736	051123	000					
3539	016741	015	051412	052105	MMADR:	.ASCIZ	<15><12>'	SET MEMORY TEST LOC IN SR'
3540	016746	046440	046505	051117				
3541	016754	040131	042524	052123				
3542	016762	046040	041517	044440				
3543	016770	020116	051123	000				
3544	016775	015	051412	052105	MMPAT:	.ASCIZ	<15><12>'	SET TEST PATTERN IN SR'
3545	017002	052040	051505	020124				
3546	017010	040520	052124	051105				
3547	017016	020116	047111	051440				
3548	017024	000122						
3549	017026	005015	046412	046505	MMPRS:	.ASCIZ	<15><12><12>'	MEMORY PARITY REGISTERS PRESENT:'<15><12>
3550	017034	051117	020131	040520				
3551	017042	044522	054524	051040				
3552	017050	043505	051511	042524				
3553	017056	051522	050040	042522				
3554	017064	042523	052116	006472				
3555	017072	000012						
3556	017074	005015	040520	044522	MTMAP:	.ASCIZ	<15><12>'	PARITY REGISTERS CONTROL MEMORY AS:'<15><12>
3557	017102	054524	051040	043505				
3558	017110	051511	042524	051522				
3559	017116	041440	047117	051124				
3560	017124	046117	046440	046505				
3561	017132	051117	020131	051501				
3562	017140	006472	000012					
3563	017144	020040	020040	000	MTYCOR:	.ASCIZ	'	'
3564	017151	055	000040		MOASH:	.ASCIZ	'	'
3565	017154	005015	000		MCR:	.ASCIZ	<15><12>	
3566	017157	113	000		MK:	.ASCIZ	'	'K'

3567	017161	116	020117	040520	MT:	.ASCIZ	'NO PARITY MEMORY FOUND'<15><12>
3568	017165	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>'CCMFAFD 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	042524	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 8K 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	046517	046120	
3624	017632	052105	000175		
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	MPCOR: .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	MPMOS: .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		017772			.EVEN
3643		000001			.END

ADRPT	000516	425#	1671*	1673*	1680*	1687	1689	1712	1802*	1902*	1904*	1911*	1916	1919
		1932	2763*	2777	2822*									
ADRS =	077400	364#												
ADRTYP	001230	660#	2655*	2668	2696	2701*	2702	2711*	2741*					
ADTEMP	015670	3283	3309#											
AE =	000001	362#	1040	1283	1335	1498	1599	1696	1812	1939	1968	1986	2128	2145
		2163	2315											
BDCNV	015524	2782	2797	3278#										
BDCNVA	015556	3285#	3288											
BDCNVB	015610	3290*	3292#											
BDCNVC	015612	3281*	3293#											
BDCNVD	015614	3282*	3289	3294#										
BEGIN	002106	692	806	815#										
BITPT	000520	426#	822*	1668	1670*	1674	1676	1681*	1682	1789*	1790	1793*	1799*	1899
		1901*	1905	1907	1912*	1913	2002	2089*	2098	2101*	2107*	2179	2262*	2597*
		2606*	2608*	2615	2626*	2633*	2634	2638	2649*	2690	2692	2707	2710*	2712
		2728*	2730*	2738	2764*	2777	2808*	2823*	2831*					
BIT0 =	000001	346#												
BIT1 =	000002	347#												
BIT10 =	002000	356#												
BIT11 =	004000	357#												
BIT12 =	010000	358#												
BIT13 =	020000	359#	2901											
BIT14 =	040000	360#	402											
BIT15 =	100000	361#												
BIT2 =	000004	348#												
BIT3 =	000010	349#												
BIT4 =	000020	350#												
BIT5 =	000040	351#	1356	1371										
BIT6 =	000100	352#	1441											
BIT7 =	000200	353#												
BIT8 =	000400	354#	2567											
BIT9 =	001000	355#	983											
BMOVE	015720	3324#	3326											
BMOVE	015710	3291	3320#											
CACHE	001224	658#	818*	910*										
CACHFL	001222	657#	737*	816	908									
CKBKO	010600	2320	2331#											
CLRPAR	014012	966	999	1090	1144	1273	1308	1402	1468	1571	1663	1705	1781	1821
		1918	1925	2087	2275	2304	2324	2416	2519	2549	2750	2954#	3106	3147
		3191												
CNT16	007314	2052#												
CNT17	010206	2157	2239#											
CNVCTR	015662	3284*	3287*	3306#										
CN16	007334	1980	2063#											
CONT	011712	2561	2567*											
CONT10	004274	1344	1350	1381#										
CONT12	005024	1510	1552#											
CONT13	005304	1609	1644#											
CONT3	002616	971#	987											
CONT4	003074	1052#	1062	1071	1078									
COUNT	004336	1337*	1351*	1391#										
DDISP =	177570	367#	616											
DECVAL	015702	3279	3314#											
DIGIT	015664	3297*	3300*	3303*	3304	3307*								
DISPLA	001102	616#	732*	832*	884*	963*	997*	1089*	1143*	1272*	1307*	1401*	1467*	1570*

CCMFAFD, 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

SEG 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)

