

PDP11-70/74

11/70 CACHE #2  
CEKBDE0

AH-7972E-MC  
FICHE 1 OF 2

MAY 1980  
COPYRIGHT © 75.80  
MADE IN USA



PDP11-70/74

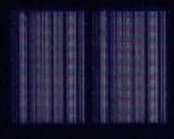
11/70 CACHE #2  
CEKBDE0

AH-7972E-MC  
FICHE 2 OF 2

MAY 1980  
COPYRIGHT © 75-80  
MADE IN USA



A large grid of approximately 10 columns and 15 rows of small, illegible text or data points, possibly representing a cache or data table. The text is too small to be read accurately.



IDENTIFICATION

PRODUCT CODE: AC-7971E-MC  
PRODUCT NAME: CEKBDE0 11/70 CACHE #2  
DATE: MAY, 1980  
MAINTAINER: DIAGNOSTIC GROUP

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

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

COPYRIGHT (C) 1975,1980 BY DIGITAL EQUIPMENT CORPORATION

## CONTENTS

1. ABSTRACT
2. REQUIREMENTS
  - 2.1 EQUIPMENT
  - 2.2 STORAGE
  - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
  - 3.1 METHOD
4. STARTING PROCEDURE
  - 4.1 CONTROL SWITCH SETTINGS
  - 4.2 STARTING ADDRESS
  - 4.3 PROGRAM AND OPERATOR ACTION
  - 4.4 SPECIAL OPERATOR INTERVENTION OPTIONS
5. OPERATING PROCEDURE
  - 5.1 OPERATIONAL SWITCH SETTINGS
  - 5.2 SUBROUTINE ABSTRACTS
  - 5.3 OPERATOR ACTION
6. ERRORS
  - 6.1 ERROR HALTS AND DESCRIPTION
  - 6.2 ERROR RECOVERY
7. RESTRICTIONS
  - 7.1 STARTING RESTRICTIONS
  - 7.2 OPERATING RESTRICTIONS
8. MISCELLANEOUS
  - 8.1 EXECUTION TIME
  - 8.2 STACK POINTER
  - 8.3 PASS COUNT
  - 8.4 ITERATIONS
  - 8.5 OSCILLOSCOPE SYNC POINTS
  - 8.6 RESTORING LOADER OR MONITOR
  - 8.7 OPTIONAL POWER DOWN POWER UP TEST
  - 8.8 MEMORY MANAGEMENT RESTRICTIONS/OPTIONS
  - 8.9 CRITICAL DEPENDENCE OF SOME TESTS ON THE  
CACHE REGISTERS
9. PROGRAM DESCRIPTION
  - 9.1 CEKBD
10. LISTINGS
  - 10.1 CEKBD

REVISION HISTORY

\*\*\*\*\*  
REV E0 1)PROGRAM MADE APT COMPATIBLE  
2)DIAGNOSTIC UTILIZATION OF MAP REGISTERS 0-2 RESTRICTED TO  
PASSIVE RELOCATION TO PREVENT APT/ACT INTERFERENCE  
3)TYPING CONTROL-C WHILE IN AUTO MODE WILL RETURN CONTROL TO  
THE MONITOR RATHER THAN HALTING PROGRAM  
4)MEMORY SIZE ROUTINE WILL NOT ACCESS MORE THAN 1920K OF MEMORY  
TO PREVENT TEST 6 FAILURE ON SYSTEMS WITH >1920K  
\*\*\*\*\*

## 1. ABSTRACT

THE PROGRAMS, CEKBC AND CEKBD, ARE INTENDED TO BE USED AS AIDS FOR THE REPAIR AND MAINTENANCE OF THE CACHE MEMORY SYSTEM IN THE PDP 11/70-74MP COMPUTING SYSTEM. THE AIM IS TO DETECT AND REPORT FAILING COMPONENTS OF THE CACHE UNIT. THE FAILURES ARE TYPICALLY IDENTIFIED WITH A FAILING CIRCUIT WHEN THE REPORT IS MADE, BUT THE OVERALL DIAGNOSTIC PHILOSOPHY HAS BEEN TO LOCATE THE FAILING MODULE (HEX BOARD) OF WHICH THERE ARE FOUR (4) IN THE CACHE UNIT. NOTE THAT WHEN A FAILURE IS REPORTED AND THE ASSOCIATED CIRCUIT IDENTIFIED, THAT CIRCUIT SHOULD NOT BE TAKEN IN BLIND FAITH AS THE DEFECTIVE COMPONENT; THE IDENTIFIED COMPONENT SHOULD RATHER BE TAKEN AS THE PROBABLE CAUSE OF THE FAILURE. THERE ARE FOUR (4) MODULES (HEX BOARDS) IN THE CACHE UNIT:

CCB	CACHE CONTROL BOARD
CDP	CACHE DATA PATHS BOARD
ADM	CACHE ADDRESS MEMORY BOARD
DTM	CACHE DATA MEMORY BOARD

THE PROGRAM CEKBC IS DESIGNED TO TEST THE FIRST TWO OF THESE BOARDS, WHILE CEKBD IS DESIGNED TO TEST THE LAST TWO BOARDS.

NOTE THAT THOUGH THE TESTING HAS BEEN DIVIDED INTO TWO STAND ALONE PROGRAMS, EACH ASSOCIATED WITH TWO MODULES, IT SHOULD NOT BE ASSUMED THAT A PARTICULAR MODULE IS WORKING AFTER HAVING RUN ONLY ONE OF THE PROGRAMS! BOTH PROGRAMS SHOULD BE RUN! FOR EXAMPLE, JUST RUNNING CEKBC WITHOUT ERROR DOES NOT RULE OUT A FAULTY COMPONENT ON THE CCB (CACHE CONTROL) BOARD.

TESTING HAS BEEN DIVIDED INTO TWO PROGRAMS ONLY BECAUSE OF THE RESTRICTIONS OF CORE SIZE RATHER THAN TO PROVIDE A MEANS OF TESTING TWO OF THE BOARDS WITH ONE PROGRAM AND THE OTHER TWO BOARDS WITH A SECOND PROGRAM. NOTE THAT CEKBD IS DESIGNED TO RUN AFTER CEKBC. IF THIS HIERARCHY IS NOT HEEDED, THAT IS IF CEKBD IS RUN BEFORE CEKBC, THEN THE ERROR REPORTING FROM CEKBD SHOULD NOT BE STRICTLY INTERPRETED.

THIS DIAGNOSTIC SUPPORTS THE KB11-B/C, AND KB11-CM PROCESSORS.

## 2. REQUIREMENTS

2.1 EQUIPMENT - PDP 11/70 CPU WITH OPERATORS CONSOLE LA30 OR EQUIVALENT TERMINAL.

2.2 STORAGE-BOTH PROGRAMS, CEKBC AND CEKBD, EACH REQUIRE 13K TO LOAD, BUT THEY BOTH ALSO ASSUME THAT THERE IS A MINIMUM OF 28K OF MEMORY IN WHICH TO RUN TESTS.

2.3 PRELIMINARY PROGRAMS - THIS PROGRAM ASSUMES THAT THE CPU IS FUNCTIONAL! THIS COULD IN SOME CIRCUMSTANCES MEAN THAT THE CPU DIAGNOSTICS SHOULD BE RUN BEFORE EITHER OF THESE DIAGNOSTICS. BUT A FAULTY MEMORY SYSTEM MAY PRECLUDE THIS, SO SITUATIONAL JUDGEMENT MUST BE USED. IF THE CPU IS KNOWN TO BE WORKING THEN RUN THESE DIAGNOSTICS, CEKBC AND CEKBD, FIRST. BUT IF THE CPU CAN NOT BE ASSUMED TO BE WORKING THEN TRY TO RUN THE CPU DIAGNOSTICS FIRST. THEN RUN THESE PROGRAMS IN ORDER: CEKBC BEFORE CEKBD! IN FACT CEKBD ASSUMES THAT MUCH OF WHAT IS TESTED IN CEKBC IS OPERATIONAL FOR DOING ITS FAULT ANALYSIS.

NOTE: THIS DIAGNOSTIC SUPPORTS THE PDP-11/74, AN EXPERIMENTAL, IN-HOUSE PROCESSOR.

### 3. LOADING PROCEDURE

3.1 METHOD - BOTH CEKBC AND CEKBD ARE LOADED FROM THE XXDP MEDIA. REFER TO THE XXDP MANUAL FOR FURTHER INFORMATION.

### 4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS (SEE 5.1)

4.2 STARTING ADDRESS - 200

4.3 PROGRAM AND OPERATOR ACTION - BOTH PROGRAMS CAN BE STARTED BY:

- 1 LOAD PROGRAM INTO MEMORY
- 2 LOAD ADDRESS 200
- 3 PRESS START
- 4 THE PROGRAMS WILL LOOP UNTIL THE HALT SWITCH IS PRESSED OR UNTIL THE USER STRIKES (TYPES) CONTROL-C (^C) ON THE TELETYPE OR TERMINAL (SEE 8.6 AND 5.2.7).

4.4 SPECIAL OPERATOR INTERVENTION OPTIONS - IF SWITCH 12 OF THE SWITCH REGISTER IS ON, THEN CEKBD WILL REQUIRE THE OPERATOR TO POWER THE MACHINE FIRST DOWN AND THEN UP (SEE 5.1 AND 8.7).

### 5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS FOR CEKBC:

SW<15>=1	HALT ON ERROR
SW<14>=1	LOOP ON TEST
SW<13>=1	INHIBIT ERROR TYPINGS
SW<12>	NOT USED IN CEKBC
SW<11>=1	INHIBIT ITERATIONS
SW<10>=1	RING BELL ON ERROR
SW<9>=1	LOOP ON ERROR
SW<8>=1	LOOP ON TEST IN SW<6:0>
SW<7>=1	SKIP EXECUTION OF TESTS WHICH USE MEMORY MANAGEMENT.
SW<6:0>	TEST NUMBER FOR LOOPING WHEN SW<8>=1

CEKBD USES THE SAME SWITCH SETTINGS AS CEKBC EXCEPT:

SW<12> =1        RUN THE OPERATOR INTERVENTION NEEDED  
POWER UP TEST

5.2        SUBROUTINE ABSTRACTS - BOTH CEKBC AND CEKBD  
USE THE FOLLOWING SUBROUTINES.

5.2.1     SPURIOUS ERROR HANDLERS - THESE ARE TWO  
ROUTINES WHICH ARE CALLED BY UNEXPECTED TRAPS TO  
EITHER VECTOR 4, IN THE CASE OF A CPU ERROR, OR  
VECTOR 114, IN CASE OF A MEMORY PARITY ERROR. THE  
CPU ERROR HANDLER, CPSPUR, TYPES OUT THE PC AT THE  
TIME OF THE TRAP AND THE CONTENTS OF THE CPU ERROR  
REGISTER (CPUERR) AND SKIPS TO THE TEST FOLLOWING  
THE ONE DURING WHICH THE ERROR OCCURRED. THE PARITY  
ERROR HANDLER, SPUR, TYPES OUT THE PC AT THE TIME OF  
THE TRAP AND THE CACHE ERROR REGISTERS, MEMERR,  
LOADRS AND HIADRS. IT THEN GIVES CONTROL TO THE  
TEST FOLLOWING THE ONE DURING WHICH THE ERROR  
OCCURRED.

5.2.2     SCOPE - THIS SUBROUTINE IS CALLED (VIA AN IOT  
INSTRUCTION) AT THE BEGINNING OF THE EXECUTION OF  
ALL THE TESTS. IT CONTROLS THE OPERATIONAL  
FUNCTIONS OF LOOPING ON TEST, ITERATION, AND SETING  
UP FOR LOOPING ON ERRORS.

5.2.3     ERROR - THIS SUBROUTINE IS CALLED (VIA AN EMT  
INSTRUCTION) TO TYPE OUT AN ERROR REPORT. IT  
CONTROLS THE OPERATIONAL FUNCTIONS OF HALTING ON  
ERROR, INHIBITING ERROR PRINT OUT, LOOPING ON ERROR,  
BELL ON ERROR, ETC.

5.2.4     TRAP CATCHER - THIS CONSISTS OF A '.+2'  
FOLLOWED BY A HALT INSTRUCTION REPEATED FROM LOCATION  
0 THROUGH 776 FOR THE PURPOSE OF CATCHING ANY  
SPURIOUS TRAP TO A VECTOR. SUCH A TRAP WILL RESULT  
IN A HALT AT THE TRAP VECTOR ADDRESS PLUS TWO (2).

5.2.5     TRAP - A NUMBER OF SUBROUTINES ARE CALLED BY  
USING THE TRAP INSTRUCTION:  
TYPE        TC TYPE OUT AN ASCII STRING  
TYPEOC     TO TYPE OUT THE OCTAL FOR A 16-BIT BINARY  
NUMBER ETC.

5.2.6     POWER DOWN AND POWER UP - THIS SUBROUTINE IS  
CALLED WHEN AN UNEXPECTED POWER DOWN OCCURS. WHEN  
POWER IS RETURNED (IF THE HALT SWITCH IS NOT ON) THE  
PROGRAM WILL RESTART AFTER TYPING A MESSAGE.

5.2.7     MONITOR OR LOADER RESTORE - WHEN THIS PROGRAM  
IS FIRST STARTED IT SAVES THE CONTENTS OF THE  
HIGHEST 1.5 (DEC) K OF MEMORY IN THE FIRST 28K.  
THESE LOCATIONS USUALLY CONTAIN THE LOADER OR  
MONITOR OF THE SYSTEM. TO RESTORE THIS LOADER OR  
MONITOR THE USER NEED ONLY TYPE CONTROL C (^C) ON

THE TERMINAL AND THAT MONITOR OR LOADER WILL  
AUTOMATICALLY BE RESTORED. AFTER THIS IS DONE THE



PROGRAM WILL HALT. NOTE THAT MANY OF THESE TESTS WIPE OUT THE ORIGINAL CONTENTS OF THAT PART OF MEMORY THEREFORE THE USER SHOULD TYPE CONTROL-C (^C) TO RESTORE THESE LOCATIONS AND AVOID HAVING TO RELOAD HIS MONITOR OR LOADER.

5.3 OPERATOR ACTION - ONLY THE POWER UP INVALIDATOR TEST IN PROGRAM CEKBD REQUIRES OPERATOR INTERVENTION, IN THE FORM OF POWERING THE PROCESSOR FIRST DOWN AND THEN UP. THIS TEST IS RUN ONLY IF SW<12>=1 (SEE 4.4 AND 5.1).

## 6. ERRORS

6.1 ERROR HALTS - ONLY TEST NUMBER 14 IN PROGRAM CEKBC, THE MAINTENANCE REGISTER COUNT PATTERN TEST, HALTS THE PROCESSOR IN THE SITUATION WHERE IT CAN'T CLEAR THE MAINTENANCE REGISTER. HERE PROCEEDING WITH THE PROGRAM'S EXECUTION WOULD PROBABLY BE FATAL, SO A HALT IS EXECUTED! NO OTHER TEST IN EITHER PROGRAM SHOULD HALT UNDER ANY NORMAL ERROR DETECTION.

6.2 ERROR RECOVERY - IF NONE OF THE ERROR PERTAINENT OPERATIONAL SWITCHES ARE BEING USED THE PROGRAM WILL EITHER RESUME THE TEST THAT MADE THE ERROR CALL OR START EXECUTION OF THE TEST FOLLOWING THE TEST DURING WHICH THE ERROR CALL WAS MADE DEPENDING ON WHETHER OR NOT THE ERROR WHICH WAS DETECTED (OR EVEN THE ERROR CALL ITSELF) WAS FATAL TO THE TEST WHICH MADE THE ERROR CALL. IF THE HALT DESCRIBED IN 6.1 ABOVE IS EVER EXECUTED THE USER CAN RESUME, IF HE IS BRAVE, BY HITTING THE CONSOLE CONTINUE SWITCH. IF ANY OF THE PERTAINENT CONSOLE SWITCH SETTING ARE SET SEE SECTION 5.1 FOR A DESCRIPTION OF THE ACTION TAKEN WHEN AN ERROR CALL IS MADE.

## 7. RESTRICTIONS

7.1 STARTING RESTRICTIONS - NONE

7.2 OPERATING RESTRICTIONS - THE MONITOR OR LOADER (OR WHAT EVER IS IN THE FIRST 28K OF MEMORY FROM LOCATIONS 152000 THROUGH LOCATION 157776) ARE SAVED SO THAT THE USER CAN RESTORE HIS LOADER OR MONITOR BY TYPING CONTROL-C (^C), (SEE 4.3 AND 5.2.7). IF THE PROGRAM WAS CHAINED IN BY A MONITOR WHICH WANTS CONTROL AUTOMATICALLY PASSED BACK TO IT WHEN TESTING IS DONE THAT MONITOR IS RESTORED AND CONTROL IS GIVEN TO IT BY THE END OF PASS ROUTINE .SEOP.

## 8. MISCELLANEOUS

8.1 EXECUTION TIME - FIRST PASS UNDER 10 SECONDS FOR BOTH PROGRAMS. SUBSEQUENT PASSES UNDER 2 MINUTES FOR BOTH PROGRAMS. (MORE EXACT EXECUTION TIMES WILL BE LATER SUPPLIED).

8.2 STACK POINTER - IN BOTH PROGRAMS THE STACK POINTER (R6) WILL BE INITIALIZED TO LOCATION 1100.

8.3 PASS COUNT - BOTH PROGRAMS WILL TYPE OUT THE PASS COUNT AT THE END OF EACH PASS.

8.4 ITERATIONS - EACH TEST HAS BEEN ASSIGNED AN ITERATION COUNT WHICH WILL DESIGNATE HOW MANY TIMES THAT TEST IS TO BE EXECUTED ON EACH PASS. NOTE THAT ON THE FIRST PASS THE ITERATION COUNT IS OVERRIDEN BY A ONE (1) MAKING ITERATIONS MEANINGLESS ON THAT FIRST PASS.

8.5 OSCILLOSCOPE SYNC POINTS - WHENEVER POSSIBLE EACH TEST HAS BEEN GIVEN AN OSCILLOSCOPE SYNC POINT (A NOP INSTRUCTION). THE ADDRESS OF THE CONDITION CODE ROM STATE (44) IS PUT IN THE PROCESSOR MICROBREAK REGISTER (177770). THIS WILL RESULT IN PIN AE1 (SLOT 10) ON THE BACK PLANE TO GO HIGH WHENEVER THE CPU ROM FLOW GOES THROUGH THE MICRO CODE ADDRESS 144. THEREFORE BY USING THE OUTPUT OF THIS BACKPLANE PIN AS A SCOPE SYNC, AND BY PUTTING A NOP INSTRUCTION IN CRUCIAL PARTS OF A TEST, THE USER WILL HAVE A VERY CONVENIENT SYNC FOR MANY SIGNALS HE MAY WISH TO OBSERVE. THE LIMITATIONS OF THIS PROCEDURE ARE THAT THE USER MUST BE ABLE TO JUDGE (DETERMINE) HOW SOON AFTER THE NOP IN THE PARTICULAR TEST HE IS RUNNING (LOOPING ON) THE SIGNAL HE WISHES TO OBSERVE SHOULD OCCUR. IN MANY CASES THIS WILL BE EASY (E.G. THE ERROR REGISTER TESTS.) BUT IN SOME TESTS THE NOP IS SO FAR FROM THE EXPECTED OCCURRENCE OF THE DESIRED SIGNAL THAT THE PROBLEM BECOMES NONTRIVIAL AND THE EXPERIENCED USER WOULD DO WELL TO FIND OTHER SYNC SIGNALS ORIGINATING IN THE CACHE DEVICE ITSELF TO OBSERVE THE LOGIC.

8.6 RESTORING THE MONITOR OR LOADER - FOR THE USERS CONVENIENCE BOTH PROGRAMS SAVE EITHER THE MONITOR OR LOADER (OR WHATEVER IS IN THE HIGHEST 1.5K OF MEMORY'S FIRST 28K) AND RESTORES IT WHEN THE USER TYPES CONTROL-C (^C) ON THE TELETYPE OR TERMINAL. THE PROGRAM, WHEN IT GETS THE CONTROL-C RESTORES THE MONITOR AND THEN HALTS. AT THIS POINT THE USERS CAN EITHER RESTART THE MONITOR OR REUSE THE LOADER ETC.

8.7 POWER UP LOGIC TEST - THERE IS A CERTAIN PART OF THE CACHE DEVICE WHICH REQUIRES A POWER DOWN POWER UP SEQUENCE TO TEST. THIS TEST HAS BEEN INCLUDED HERE AS AN OPTION ONLY BECAUSE IT REQUIRES OPERATOR INTERVENTION. TO RUN THIS TEST SET SW<12>=1 (CEKBD ONLY. SEE 5.1).

8.8 MEMORY MANAGEMENT RESTRICTIONS/OPTIONS - MANY OF THE TESTS REQUIRE THE USE OF EXTENSIVE MEMORY MANAGEMENT MAPPING FACILITIES. THESE TESTS MUST ASSUME THE MEMORY MANAGEMENT (AND SOME OF THE MAPPING BOX) IS OPERATIONAL. NORMALLY THESE TEST WILL BE EXECUTED. BUT THE FEATURE HAS BEEN PROVIDED WHEREBY THE USER CAN DELETE THE EXECUTION OF ANY TESTS WHICH REQUIRE THE USE OF MEMORY MANAGEMENT AND/OR THE MAPPING. THIS HAS BEEN IMPLEMENTED USING SW<7>. WHEN THIS SWITCH IS 0 NORMAL OPERATION IS UNDERTAKEN, BUT WHEN SW<7>=1 THEN ANY TEST WHICH MUST TURN ON THE MEMORY MANAGEMENT UNIT (THE MAPPING BOX) WILL NOT BE RUN AND CONTROL WILL BE PASSED TO THE NEXT TEST!

8.9 CRITICAL DEPENDENCE OF SOME TESTS ON THE CACHE REGISTERS - AS THE PROGRAMS RUN, FLAGS ARE SET WHICH DESIGNATE THE FUNCTIONALITY OF A CACHE REGISTER. IF A TEST DETERMINES THAT A PARTICULAR REGISTER IS NOT FUNCTIONAL IT SETS A FLAG WHICH DESIGNATES TO THE REST OF THE PROGRAM THAT THAT REGISTER DOES NOT WORK PROPERLY. SOME TESTS WHICH RELY ON THE REGISTERS TO BE FUNCTIONAL WILL TEST THESE FLAGS AND IF THEY FIND THEM TO INDICATE THAT A REGISTER THEY NEED IS BAD THEY WILL SKIP TO THE NEXT TEST!

## 9. PROGRAM DESCRIPTION

### 9.1 CEKBD

COPYRIGHT 1975, 1979 DIGITAL EQUIPMENT CORPORATION MAYNARD, MASS. 01754

COPYRIGHT (C) 1975, 1979 DIGITAL EQUIPMENT CORP. MAYNARD, MASS. 01754

PROGRAM BY ANTHONY S. VEZZA

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC PACKAGE (MAINDEC-11-DZQAC-A5-1).

#### TEST 1 PARITY ERROR ABORT

THIS TEST ENSURES THAT A CACHE PARITY ERROR FLAG CAUSES AN ABORT. THIS IS DONE BY FORCING A PARITY ERROR ON AN EVEN WORD.

#### TEST 2 PARITY ERROR TRAP

THIS TEST ENSURES THAT A PARITY TRAP FUNCTIONS PROPERLY. THIS IS DONE BY MAKING THE ODD WORD HAVE BAD PARITY. IF THE TRAP DOES'T OCCUR THEN THE PROBLEM IS ON TMCA. IF A TRAP OCCURS TO THE WRONG VECTOR THE PROBLEM COULD BE ON TMCA OR UCB.

TEST 3 MEM MGT AND PE TRAP PRIORITY ARBITRATION

K 1

SEQ 0010

THIS TEST ENSURES THAT THE ARBITRATION LOGIC WORKS FOR MEMORY MANAGEMENT AND PARITY ERROR TRAPS.

TEST 4 UNIBUS PARITY ERROR

THIS TEST MAKES A REFERENCE TO MEMORY THRU MAPPING BOX THAT WILL CAUSE A PARITY ERROR. IF ABORT DOESN'T HAPPEN THEN THE PROBLEM IS ON UBCB.

NOTE: MAP REGISTER 0 AND 1 ARE NOT USED INCASE THE PROGRAM IS RUNNING UNDER ACT11.

TEST 5 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ONES

THIS TEST IS A TEST OF BOTH THE AMX, CPU INPUTS, AND THE CACHE ERROR ADDRESS REGISTER. A SET OF ADDRESSES IS GENERATED AND A MAIN MEMORY ADDRESS AND CONTROL LINE PARITY ERROR IS FORCED AT EACH, THEREBY LOCKING UP THE ADDRESS ON THE OUTPUT OF THE AMX IN THE ERROR ADDRESS REGISTER. THE MANNER IN WHICH THIS IS DONE IS AS FOLLOWS: FIRST THE ADDRESS IS GENERATED; THEN, IF IT IS A VALID ADDRESS (THAT IS, IF IT IS NOT BEYOND THE LIMITS OF MEMORY AS DISPLAYED IN THE SYSTEM SIZE REGISTER), THESE THREE INSTRUCTIONS ARE MOVED TO THAT AREA OF MEMORY:

ONE: MOV R1,(R2)  
2\$: CLR (R2)  
3\$: RTS PC 2\$ IS THE ADDRESS BEING TESTED. THE INSTRUCTION AT ONE IS GIVEN CONTROL BY A 'JSR PC'. R1 IS MADE TO CONTAIN #2 AND R2 CONTAINES THE ADDRESS OF THE MAINTENANCE REGISTER, SO THAT AFTER THE 'MOV R1,(R2)' IS EXECUTED A PARITY ERROR SHOULD OCCUR ON THE MAIN MEMORY ADDRESS AND CONTROL LINES WHEN THE NEXT INSTRUCTION IS FETCHED. THE ADDRESSES USED ARE GENERATED FOLLOWING THIS PATTERN

200000 200002 200004  
200010 200020 200040  
200100 200200 200400  
ETC. TO: 240000  
300000 400000 400002  
400004 400010 ETC.  
TO: 500000 600000  
1000000 1000002  
1000004 ETC.

THE PATTERN CONTINUES UNTIL AN ADDRESS IS GENERATED THAT IS TOO LARGE. MEMORY MANAGEMENT IS SET UP TO FULL 22-BIT MODE, SO IF THE USER WANTS TO HAVE THE EXECUTION OF THIS TEST DELETED HE CAN SIMPLY BY TURNING ON THE APPROPRIATE CONSOLE SWITCH WHICH HAS BEEN DESIGNATED FOR THE PURPOSE OF DELETING THE EXECUTION OF TESTS WHICH MAKE USER OF MEMORY MANAGEMENT.

TEST 6 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ZEROES

THIS IS ANOTHER TEST OF THE AMX WHICH IS CARRIED OUT USING THE SAME METHOD AS IN THE PREVIOUS TEST ALL THAT IS DIFFERENT IS THE SERIES OF TEST ADDRESSES WHICH IS USED. IN THE PREVIOUS TEST A ONE WAS FLOATED THROUGH A FIELD OF ZEROES TO PRODUCE THE TEST ADDRESSES, HERE A ZERO WILL BE FLOATED THROUGH A FIELD OF ONES TO PRODUCE THE ADDRESSES BASE ADDRESSES WHICH ARE USE ARE:

```

177776 377776 777776
1777776      3777776
7777776 1777776

```

EACH OF THESE PATTERNS IS TAKEN AND A ZERO IS FLOATED THROUGH THE FIELD OF ONES TO PRODUCE A TEST ADDRESS.

TEST 7 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ONES

THIS IS A TEST OF THE UNIBUS INPUTS TO THE AMX. THIS TEST IS IDENTICAL TO TST1 IN EVERY THING IT DOES EXCEPT IN THAT TEST THE TEST ADDRESSES WERE REFERENCED THROUGH MEMORY MANAGEMENT STRAIGHT FROM THE CPU TO THE CACHE. HERE THE TEST ADDRESSES WILL GO THROUGH THE MEMORY MANAGEMENT UNIT ONTO THE UNIBUS WHERE THE MAPPING BOX WILL SEND THEM TO THE CACHE AS UNIBUS REFERENCES.

TEST 10 CACHE ADDRESS MULTIPLEXER, AMX,  
UNIBUS INPUTS TEST FLOATING ZEROES

THIS IS A TEST OF THE UNIBUS INPUTS TO THE AMX. THIS TEST IS IDENTICAL TO TST2 IN EVERY THING IT DOES EXCEPT IN THAT TEST THE TEST ADDRESSES WERE REFERENCED THROUGH MEMORY MANAGEMENT STRAIGHT FROM THE CPU TO THE CACHE. HERE THE TEST ADDRESSES WILL GO THROUGH THE MEMORY MANAGEMENT UNIT ONTO THE UNIBUS WHERE THE MAPPING BOX WILL SEND THEM TO THE CACHE AS UNIBUS REFERENCES.

TEST 11 CACHE ADDRESS MULTIPLEXER, AMX, CPU  
INPUTS DUAL ADDRESS TEST

THIS TEST PERFORMS A DUAL ADDRESS TEST ON MEMORY LOCATED AT ADDRESSES LESS THAN 160000 (OCT.) OR WITHIN THE FIRST 28K. THE PURPOSE IS TO VERIFY THE THE AMX IS WORKING PROPERLY FOR THE LOW ORDER ADDRESS LINES INVOLVED.

TEST 12 CACHE ADDRESS MULTIPLEXER, AMX,  
UNIBUS INPUTS DUAL ADDRESS TEST

THIS TEST PERFORMS A DUAL ADDRESS TEST IDENTICAL TO TST5, EXCEPT THAT IT IS DONE THROUGH THE MAPPING BOX HERE THEREBY TESTING THE UNIBUS INPUTS TO THE AMX.

TEST 13 CACHE ADDRESS MEMORY COMPARATOR TEST

THIS IS A TEST OF THE CACHE ADDRESS MEMORY ADDRESS COMPARATORS. THIS IS A CIRCUIT MADE UP OF SIX 74585 CHIPS, THREE FOR EACH GROUP. EACH CHIP COMPARES FOUR BITS OF THE ADDRESS ON THE ADDRESS MULTIPLEXER, AMX, OUTPUT LINES WITH THE RESPECTIVE FOUR BITS FROM THE CACHE ADDRESS MEMORY. TWELVE BITS OF THE ADDRESS ARE BROKEN DOWN THUS: BITS 10 THROUGH 13 FOR THE FIRST COMPARATOR; BITS 14 THROUGH 17 FOR THE NEXT; AND BITS 18 THROUGH 21 FOR THE LAST. THE METHOD CHOSEN FOR THIS TEST IS TO TAKE EACH POSSIBLE 4-BIT INPUT CONDITION FOR A COMPARATOR FROM THE ADDRESS MEMORY AND PUT EVERY POSSIBLE 4-BIT COMBINATION ON THE AMX SIDE OF THE COMPARATOR. FOR 4-BITS THERE ARE 16

(DEC) CONDITIONS. THUS FOR EVERY 4-BIT ADDRESS MEMORY INPUT TO THE COMPARATOR THERE ARE 16 AMX INPUT COMBINATIONS ONE OF WHICH WILL CAUSE A MATCH AND MAKE THE REFERENCE A HIT. THE OTHER 15 SHOULD OF COURSE BE MISSES.

TEST 14 CACHE ADDRESS MEMORY COUNT PATTERN TEST

THIS IS A TEST OF THE ADDRESS MEMORY IN THE CACHE. EVERY BIT IN THE MEMORY IS TURNED ON AND OFF WITHIN THE LIMITATIONS OF MEMORY SIZE. THE MANNER IN WHICH THIS IS DONE IS TO ATTEMPT TO MAKE EVERY ADDRESS IN AVAILABLE MEMORY A HIT IN EACH GROUP.

TEST 15 CACHE ADDRESS MEMORY PARITY LOGIC TEST

THIS IS A TEST OF THE PARITY CHECKERS AND PARITY GENERATOR OF THE CACHE ADDRESS MEMORY. EVERY POSSIBLE ADDRESS TAG, BITS 21 THROUGH 10, WHICH CAN BE STORED IN THE CACHE ADDRESS MEMORY IS GENERATED, MADE A HIT AND THE MAINTENANCE REGISTER IS THEN USED TO FORCE A CACHE ADDRESS MEMORY PARITY ERROR AT EACH OF THE ADDRESSES GENERATED. NOTE THAT BITS 9 THROUGH 0 OF THE ADDRESSES IS NOT OF CONCERN, SO THESE BITS WILL BE THE SAME FOR EACH ADDRESS; THIS IS BECAUSE ONLY BITS 21 THROUGH 10 ARE STORED IN THE ADDRESS MEMORY THEREFORE ONLY THESE BITS ARE PARITY CHECKED IN THE CACHE ADDRESS MEMORY PARITY CHECKERS. ALSO NOTE THAT THE RANGE OF THE ADDRESSES MUST BE LIMITED TO BETWEEN THE BOUNDS IMPOSED BY THE HIGHEST AVAILABLE MEMORY WORD AND THE LAST WORD OF MEMORY USED BY THIS PROGRAM. THE MANNER IN WHICH THE ERROR WILL BE FORCED WILL BE TO PUT THE INSTRUCTIONS:

```

1$:    MOV    R4,(R2)
TSTADS: CLR   (R2)
        RTS   PC AT THE
PARTICULAR ADDRESS BEING TESTED,
WHERE 'TSTADS' IS THE ADDRESS BEING
TESTED. R4 CONTAINS A PATTERN TO BE
LOADED IN THE MAINTENANCE REGISTER

```

WHICH WILL FORCE AN ERROR IN THE CACHE ADDRESS MEMORY; R2 CONTAINS THE ADDRESS OF THE MAINTENANCE REGISTER. NOTE FOR EACH ADDRESS R4 WILL FIRST BE SUCH AS TO CAUSE AN ERROR IN THE LOW BYTE ADDRESS PARITY CHECKER THEN AT THE SAME ADDRESS AN ERROR WILL BE FORCED ON THE HIGH BYTE! THE SEQUENCE OF TEST ADDRESSES WILL BE GENERATED TWICE ONCE MAKING THEM HITS IN GROUP 0 THEN MAKING THEM HITS IN GROUP 1.

TEST 16 CACHE ADDRESS MEMORY DUAL ADDRESS TEST, UPWARD

THIS IS A DUAL ADDRESS TEST OF THE CACHE ADDRESS MEMORY. AS MANY AS POSSIBLE DIFFERENT ADDRESS 'TAGS' ARE STORED IN THE 256 (DEC) ADDRESS LOCATIONS OF THE GROUP BEING TESTED. OBVIOUSLY THE NUMBER OF DIFFERENT ADDRESS TAGS AVAILABLE IS LIMITED BY THE SIZE OF THE MEMORY ON THE SYSTEM. NOTE THAT HERE THE WORD 'TAG' REFERS TO THAT PART OF AN ADDRESS, BITS 10 THROUGH 21, WHICH ARE STORED IN THE CACHE ADDRESS MEMORY. HERE THE ADDRESS MEMORY IS WRITTEN IN THE UPWARD DIRECTION, THAT IS 'TAG' 1 IS WRITTEN FIRST, 'TAG' 2 SECOND ETC. THEN EACH ADDRESS WHICH WAS WRITTEN IS TESTED TO SEE IF IT IS A HIT, THUS MAKING SURE NO 'TAG' WAS OVERWRITTEN BY A REFERENCE TO ANOTHER 'TAG'. NOTE THAT THIS DOES NOT PERFORM A COMPLETE DUAL ADDRESS TEST ON THE ADDRESS MEMORY, FOR THAT WOULD INVOLVE WRITTING THE 'TAGS' IN THE DOWNWARD DIRECTION AS WELL AS THE UPWARD DIRECTION. THE DOWNWARD WRITING PART OF THIS DUAL ADDRESS TEST IS FOUND IN TST13.

TEST 17 CACHE ADDRESS MEMORY DUAL ADDRESS TEST, DOWNWARD

THIS IS A DUAL ADDRESS TEST OF THE CACHE ADDRESS MEMORY. AS MANY AS POSSIBLE DIFFERENT ADDRESS 'TAGS' ARE STORED IN THE 256 (DEC) ADDRESS LOCATIONS OF THE GROUP BEING TESTED. OBVIOUSLY THE NUMBER OF DIFFERENT ADDRESS TAGS AVAILABLE IS LIMITED BY THE SIZE OF THE MEMORY ON THE SYSTEM. NOTE THAT HERE THE WORD 'TAG' REFERS TO THAT PART OF AN



ADDRESS, BITS 10 THROUGH 21, WHICH ARE STORED IN THE CACHE ADDRESS MEMORY. HERE THE ADDRESS MEMORY IS WRITTEN IN THE DOWNWARD DIRECTION, THAT IS 'TAG' 256 IS WRITTEN FIRST, 'TAG' 255 SECOND ETC. THEN EACH ADDRESS WHICH WAS WRITTEN IS TESTED TO SEE IF IT IS A HIT, THUS MAKING SURE NO 'TAG' WAS OVERWRITTEN BY A REFERENCE TO ANOTHER 'TAG'. NOTE THAT THIS DOES NOT PERFORM A COMPLETE DUAL ADDRESS TEST ON THE ADDRESS MEMORY, FOR THAT WOULD INVOLVE WRITTING THE 'TAGS' IN THE UPWARD DIRECTION AS WELL AS THE DOWNWARD DIRECTION. THE UPWARD WRITING PART OF THIS DUAL ADDRESS TEST IS FOUND IN TST12.

TEST 20 CACHE ADDRESS MEMORY BYTE MASK  
GENERATOR, CPU DATOB ONES TEST

THIS IS A TEST OF THE BYTE MASK GENERATION LOGIC. THIS IS A FOUR BIT MASK USED BY MAIN MEMORY WHEN PERFORMING A WRITE. IT DESIGNATES WHICH BYTES OF THE TWO WORDS OF DATA ON THE MAIN MEMORY DATA BUS LINES ARE TO BE WRITTEN. THIS WILL BE A TEST DOING CPU DATOB REFERENCES TO THE CACHE. THE DATOB WILL WRITE 377 INTO A BACK ROUND PATTERN OF ZEROES.

TEST 21 CACHE ADDRESS MEMORY BYTE MASK  
GENERATOR, CPU DATOB ZEROES TEST

THIS IS ANOTHER TEST OF THE BYTE MASK GENERATION LOGIC. HERE CPU DATOB'S WILL MOVE ZEROES INTO A BACKGROUND PATTERN OF ONES.

TEST 22 CACHE ADDRESS MEMORY BYTE MASK  
GENERATOR, UNIBUS DATOB ONES TEST

THIS IS A TEST OF THE BYTE MASK GENERATION LOGIC. THIS IS A FOUR BIT MASK USED BY MAIN MEMORY WHEN PERFORMING A WRITE. IT DESIGNATES WHICH BYTES OF THE TWO WORDS OF DATA ON THE MAIN MEMORY DATA BUS LINES ARE TO BE WRITTEN. THIS WILL BE A TEST DOING UNIBUS DATOB REFERENCES TO THE CACHE. THE DATOB WILL WRITE 377 INTO A BACK ROUND PATTERN OF ZEROES.

TEST 23 CACHE ADDRESS MEMORY BYTE MASK  
GENERATOR, UNIBUS DATOB ZEROES TEST

THIS IS ANOTHER TEST OF THE BYTE MASK GENERATION LOGIC. HERE UNIBUS DATOB'S WILL MOVE ZEROES INTO A BACKGROUND PATTERN OF ONES.

TEST 24 CACHE ADDRESS MEMORY POWER UP  
INVALIDATOR TEST

THIS TEST IS EXECUTED OPTIONALLY, ON THE CONDITION THAT BIT 12 OF THE SWITCH REGISTER IS ON WHEN PROGRAM CONTROL REACHES THIS POINT. IF THIS SWITCH IS OFF THEN CONTROL IS PASSED TO THE NEXT TEST. THIS IS DONE BECAUSE THIS TEST REQUIRES OPERATOR INTERVENTION. THE USER IS ASKED TO GO THROUGH A POWER DOWN-POWER UP SEQUENCE. THEN A SIMPLE SCAN IS MADE OF MEMORY WHICH CAUSES ALL DATA AND ADDRESS MEMORY LOCATIONS IN THE CACHE TO BE PARITY CHECKED. IF THE POWER UP-CACHE INVALIDATOR LOGIC WORKED NO PARITY ERRORS CAN OCCUR. BUT IF THIS INVALIDATOR FAILED THERE IS AN EXTREMELY HIGH PROBABILITY FOR THE OCCURENCE OF A CACHE DATA OR CACHE ADDRESS PARITY ERROR. IN FACT IF THE INVALIDATOR CIRCUIT IS COMPLETELY INOPERATIVE IT WILL BE VIRTUALLY IMPOSSIBLE TO RESTART THE PROGRAM. WHEREAS MINOR OR NO FAILURES CAN AND WILL BE REPORTED. IF NO PARITY ERRORS ARE ENCOUNTERED THE USER WILL BE NOTIFIED SO THAT HE CAN KNOW IF A FATAL FAILURE HAS OCCURRED.

TEST 25 CACHE DATA MULTIPLEXER, CDMX, TEST

THIS TEST PUTS DIFFERENT PATTERNS OF DATA AT THE INPUTS OF THE CDMX AND TESTS FOR PROPER SELECTION AND GOOD DATA.

TEST 26 CACHE DATA MEMORY ADDRESS DRIVERS  
TEST

THIS TEST PERFORMS A DUAL ADDRESS TEST ON THE CACHE DATA MEMORIES OF BOTH GROUPS.

## TEST 27 CACHE DATA MEMORY COUNT PATTERN TEST

THIS TEST RUNS A COUNT PATTERN THROUGH EACH LOCATION OF THE CACHE DATA MEMORY FOR EACH GROUP.

## TEST 30 CACHE DATA MEMORY PARITY CHECKERS LOW BYTE TEST

THIS IS A TEST OF THE TWO CACHE DATA MEMORY PARITY CHECKERS FOR THE LOW BYTE, ONE FOR EACH GROUP. THE MAINTENANCE REGISTER IS USED TO FORCE A PARITY A PARITY ERROR AT EVERY DATA PATTERN WHICH HAS A ONE PARITY BIT. NOTE THAT THE CACHE DATA MEMORY PARITY HAS, EFFECTIVELY, ODD PARITY. THE MAINTENANCE FUNCTION ON THE CACHE DATA MEMORY PARITY CHECKERS HAS THE EFFECT OF FORCING THE PARITY BIT OF THE BYTE BEING CHECKED TO ZERO. THIS MEANS THAT ONCE THIS MAINTENANCE FUNCTION IS ENABLED THE ERROR WILL OCCUR ON A SUBSEQUENT READ OF A BYTE WITH A ONE PARITY BIT, THAT IS BYTES WITH ZERO PARITY BITS WILL NOT CAUSE THE ERROR.

## TEST 31 CACHE DATA MEMORY PARITY CHECKERS HIGH BYTE TEST

THIS IS A TEST OF THE TWO CACHE DATA MEMORY PARITY CHECKERS FOR THE HIGH BYTE, ONE FOR EACH GROUP. THE MAINTENANCE REGISTER IS USED TO FORCE A PARITY A PARITY ERROR AT EVERY DATA PATTERN WHICH HAS A ONE PARITY BIT. NOTE THAT THE CACHE DATA MEMORY PARITY HAS, EFFECTIVELY, ODD PARITY. THE MAINTENANCE FUNCTION ON THE CACHE DATA MEMORY PARITY CHECKERS HAS THE EFFECT OF FORCING THE PARITY BIT OF THE BYTE BEING CHECKED TO ZERO. THIS MEANS THAT ONCE THIS MAINTENANCE FUNCTION IS ENABLED THE ERROR WILL OCCUR ON A SUBSEQUENT READ OF A BYTE WITH A ONE PARITY BIT, THAT IS BYTES WITH ZERO PARITY BITS WILL NOT CAUSE THE ERROR.

TEST 32 CACHE DATA MEMORY WORST CASE NOISE TEST

THIS TEST DOES A GALLOPING 0'S AND 1'S OR PING PONG TEST ON THE CACHE BIPOLAR DATA MEMORY.

TEST 33 CACHE DATA MEMORY CHIP SELECTION LOGIC TEST

THIS ROUTINE TESTS THE 'CHIP-SET' ENABLE LOGIC FOR THE CACHE DATA MEMORY. TO DEFINE THE TERM 'CHIP-SET' CONSIDER THE CACHE MEMORY AS BEING DIVIDED INTO FOUR SETS OF 256 (DEC) X 1 BIT BIPOLAR MEMORY CHIPS. EACH SET IS MADE UP OF 18 CHIPS, THE 745200, EACH CHIP REPRESENTS ONE BIT OF DATA OR PARITY, THUS 16 DATA BITS PLUS TWO PARITY BITS CORRESPOND TO THE 18 CHIPS IN EACH GROUP. THE 'CHIP-SETS' THEN CORRESPOND TO THE STRUCTURE OF THE MEMORY IN THIS WAY:

SET 0 GROUP 0 EVEN WORD

SET 1 GROUP 0 ODD WORD

SET 2 GROUP 1 EVEN WORD

SET 3 GROUP 1 ODD WORD A

DIFFERENT PATTERN, 000000 177777 125252 AND 052525, IS WRITTEN INTO EACH GROUP AND THEN READ BACK. EVERY PERMUTATION OF THE FOUR TEST PATTERNS IN THE FOUR SETS IS TRIED AND CHECKED. FOR EACH PERMUTATION OF THE TEST PATTERNS THIS ROUTINE FIRST WRITES 'UP' (SET 0 FIRST THEN 1,2 AND 3) THEN 'DOWN' (SET 3 FIRST THEN 2,1 AND 0).

TEST 34 CACHE DATA MEMORY BYTE ENABLE LOGIC TEST

THIS TEST PERFORMS A CHECK OF THE BYTE ENABLE LOGIC IN THE CACHE DATA MEMORY. THE BYTE PATTERNS 1, 2, 4, 10, 20, 40, 100 A 200 ARE USED. THE FIRST FOUR PATTERNS ARE WRITTEN IN CONSECUTIVE BYTE LOCATIONS WHICH ARE HITS IN GROUP 0. THE REMAINING FOUR PATTERNS ARE WRITTEN IN CONSECUTIVE BYTE LOCATIONS WHICH ARE HITS IN GROUP 1. EACH PATTERN IS READ BACK CHECKED AND THE COMPLIMENT PATTERN IS WRITTEN. AFTER ALL THE PATTERNS HAVE BEEN CHECKED AND COMPLEMENTED THE COMPLIMENTED PATTERNS ARE CHECKED.

THIS IS A TEST OF:

1. CACHE ARBITRATION
2. THE MASS BUS AND  
UNIBUS PORTS TO THE CACHE
3. HIGH SPEED I/O  
THROUGH THE CACHE

IT MAKES USE OF THE FOLLOWING  
DEVICES:

1. RS04
2. RP04
3. RK05
4. MASS BUSS TESTER
5. UNIBUS EXERCISER

IF ANY OF THESE DEVICES ARE PRESENT  
AND WRITE ENABLED THEY WILL BE USED  
IN THIS TEST. ONLY THE LOWEST WRITE  
ENABLED DRIVE NUMBER OF EACH DEVICE  
WILL BE USED.

CAUTION!!! THIS TEST WILL  
WRITE ON THE DISKS IT USES.  
SO VITAL SYSTEMS DISKS  
SHOULD BE REMOVED OR WRITE  
PROTECTED BEFORE RUNNING  
THIS DIAGNOSTIC.

IF UNIT ZERO OF A PARTICULAR DEVICE  
IS WRITE PROTECTED THEN THIS TEST  
WILL TRY TO USE UNIT ONE, ETC.

ALL AVAILABLE DEVICES ARE STARTED  
DOING TRANSFERS AT THE SAME TIME TO  
DIFFERENT PARTS OF MEMORY. EACH  
DEVICE HAS A CONTROL ROUTINE WHICH  
DRIVES THAT DEVICE THROUGH THE  
CYCLE:

1. WRITE A RANDOM DATA  
PATTERN IN MEMORY
2. COPY THAT PATTERN  
ONTO THE DISK
3. WRITE CHECK THE DISK
4. READ THE PATTERN OFF  
THE DISK BACK INTO MEMORY
5. CHECK DATA
6. START OVER AT 1.

EACH DEVICE IS CAUSED TO GO THROUGH THIS CYCLE A PREDETERMINED NUMBER OF TIMES. THIS NUMBER IS CONTAINED IN THE LOCATION, CYCNT, AND CAN BE CHANGED BY THE USER AT THE CONSOLE TO ANY VALUE HE DESIRES.

INTERRUPTS ARE ENABLED SO THAT IT IS POSSIBLE TO GET MANY DEVICES DOING TRANSFERS AT ONCE.

UNFORTUNATELY THE DEGREE TO WHICH FAULTS CAN BE ISOLATED IS LIMITED BY THE FACT THAT THERE ARE MANY ELEMENTS, DEVICES, INVOLVED. THESE ERRORS ARE REPORTED:

1. ALL DEVICE ERRORS
2. ALL DATA OR PARITY ERRORS

NOTE THAT THIS NOT INTENDED TO BE USED AS AN I/O DEVICE DIAGNOSTIC! ALL THE DEVICES WHICH ARE USED ARE ASSUMED TO BE IN PROPER WORKING CONDITION.

#### TEST 36 MASS BUS CACHE WRITE HIT CYCLE, INVALIDATION TEST

THIS IS A TEST OF CACHE INVALIDATION ON MASS BUS CYCLES WHICH ARE WRITE HITS IN THE CACHE. A GROUP OF LOCATIONS IS MADE HITS AND THEN A MASS BUS DEVICE IS CALLED UPON TO DO TRANSFERS, WRITES TO THOSE LOCATIONS. THOSE WRITES SHOULD THUS BE INVALIDATED.

NOTE: THE FOLLOWING TESTS ARE EXECUTED ON A KB11-CM ONLY!

#### TEST 37 CHECK IVSS, VSIU BITS

THIS TEST CHECKS THAT THE IVSS AND VSIU BITS OF THE CACHE CONTROL REGISTER CAN BE SET AND CLEARED. VCIP IS ALSO CHECKED.

#### TEST 40 CHECK VSIU BIT, WITH IVSS ALREADY SET

THIS TEST CHECKS THAT THE 'VALID STORE IN USE' (VSIU) BIT CAN BE SET AND CLEARED WHEN THE IVSS IS ALREADY SET.

## TEST 41 CHECK VCIP SETS WHEN CF IS SET

THIS TEST CHECKS THAT THE VCIP SETS WHEN CACHE-FLUSH IS DONE AND IT CLEARS OUT WITHIN A CERTAIN TIME AFTER THE FLUSH OF VALID STORE IS OVER

## TEST 42 CHECK CACHE FLUSH &amp; VALID STORE SWITCHING

THIS TEST CHECKS THAT WHEN A CACHE FLUSH IS DONE BY SETTING CF IN CCR, THE VALID STORE IN USE (VSIU) SURTCHES. VALID STORE SWITCHING FROM STORE-A TO STORE-B AND VICE-VERSA IS CHECKED

## TEST 43 CHECK IVSS INHIBITS SWITCHING OF VALID STORE IN USE

THIS TEST CHECKS THAT WHEN "INHIBIT VALID STORE SWITCHING" (IVSS) IS SET AND FLUSH-CACHE BIT IS SET, THE VALID STORE IN USE DOES NOT SWITCH

## TEST 44 CHECK VALID STORES (A &amp; B) FOR GROUP 0

THIS TEST CHECKS THE TWO VALID STORES (A&B) FOR GROUP 0 OF THE CACHE. WHEN A CACHE-FLUSH IS ISSUED, THE CACHE SHOULD BE INVALIDATED BY SWITCHING THE VALID STORE IN USE THE TEST-CODE IS MADE HIT IN GROUP 1 (WHICH IS NOT BEING TESTED). THE TEST DATA IS MADE HIT IN GROUP 0. FLUSH-CACHE BIT IS SET IN THE CCR. IT IS CHECKED THAT THE TEST-DATA WHICH WAS HIT (MADE PREVIOUSLY) IN GROUP 0 IS NO MORE A HIT. EACH LOCATION OF THE TEST-DATA BLOCK IS REFERENCED AND CHECKED IF IT WAS A MISS. OTHERWISE AN ERROR IS REPORTED. AS A RESULT OF THE CACHE FLUSH THE VALID STORE SHOULD HAVE SWITCHED FROM 0 TO 1. THEN THE VALID STORE IS FORCED TO BE 0 AND THE TEST-DATA IS REFERENCED AGAIN. IT IS CHECKED IF IT WAS A MISS.

THIS TEST CHECKS THAT HIT CAN BE OBTAINED FROM BOTH GROUPS (0&1) OF THE CACHE, FROM EACH OF THE TWO VALID STORES (A&B) PER GROUP. THUS ALL 4 VALID STORES GET CHECKED. TEST-DATA (UNIQUE) IS MADE A HIT IN GROUP 0 USING THE FIRST VALID STORE A. TEST-CODE IS MADE A HIT IN THE GROUP NOT BEING TESTED. TEST-DATA IS READ BACK AND CHECKED FOR CORRECTNESS. IT IS ALSO CHECKED IF THE TEST-DATA REFERENCE WAS A HIT. THE TESTING IS REPEATED FOR VALID STORE B. THE ENTIRE TEST (ABOVE) IS REPEATED FOR GROUP 1.

## TEST 46 CHECK VALID STORES (A &amp; B ) FOR GROUP 1

THIS TEST CHECKS THE TWO VALID STORES (A&B) FOR GROUP 1 OF THE CACHE. WHEN A CACHE-FLUSH IS ISSUED, THE CACHE SHOULD BE INVALIDATED BY SWITCHING THE VALID STORE IN USE. THE TEST-CODE IS MADE HIT IN GROUP 1 (WHICH IS NOT BEING TESTED). THE TEST DATA IS MADE HIT IN GROUP 0. FLUSH-CACHE HIT IS SET IN THE CCR. IT IS CHECKED THAT THE TEST-DATA WHICH WAS HIT (MADE PREVIOUSLY) IN GROUP 0 IS NO MORE A HIT, EACH LOCATION OF THE TEST-DATA BLOCK IS REFERENCED AND CHECKED IF IT WAS A MISS. OTHERWISE AN ERROR IS REPORTED. AS A RESULT OF THE CACHE FLUSH THE VALID STORE SHOULD HAVE SWITCHED FROM 0 TO 1. THEN THE VALID STORE IS FORCED TO BE 0 AND THE TEST-DATA IS REFERENCED AGAIN. IT IS CHECKED IF IT WAS A MISS. THE WHOLE TEST IS REPEATED USING VALID-STORE B (1).

## TEST 47 CHECK CACHE TURNS OFF WHEN FLUSH IS DONE WITH IVSS SET

THIS TEST CHECKS THAT IF CACHE-FLUSH IS DONE (SETTING CF), WHEN IVSS IS SET, THE VALID STORES ARE NOT SWITCHED AND THE CACHE IS TURNED OFF (AND A SLOW FLUSH IS PERFORMED). THUS, ANY REFERENCE TO A PREVIOUSLY CACHED DATA SHOULD RESULT IN CACHE MISS. TEST-DATA IS MADE HIT IN GROUP 0 (BEING TESTED). TEST CODE IS MADE HIT IN GROUP 1. IVSS IS SET AND A FLUSH IS DONE. PREVIOUSLY CACHED TEST-DATA IS REFERENCED TO CHECK IT IS A MISS. THE TEST IS REPEATED FOR BOTH GROUPS AND VALID STORES.



TEST 50 CHECK CACHE TURNS OFF ON A BACK-TO-BACK FLUSH

K 2

SEQ 0023

THIS TEST CHECKS THAT THE CACHE TURNS OFF AND FORCES ALL REFERENCES TO THE MAIN MEMORY WHEN BACK-TO-BACK CACHE FLUSHES ARE DONE. WHEN A CACHE FLUSH IS INITIATED WHILE THE PREVIOUS ONE IS IN PROGRESS, IT IS KNOWN AS BACK-TO-BACK FLUSH.

TEST 51 CHECK CACHE-BYPASS

THIS TEST CHECKS THE CACHE BYPASS FUNCTION. WHEN THE 'BYPASS CACHE' IS SET IN THE CACHE CONTROL REGISTER ALL REFERENCES ARE FORCED TO MAIN MEMORY. IF A READ OR WRITE HIT OCCURS THAT LOCATION IS INVALIDATED IN THE TAG STORE. FIRST, THE TEST CODE IS MADE HIT IN GROUP 1 BY FORCE-REPLACING GROUP 1. THEN THE TEST-DATA IS MADE HIT IN GROUP 0. CACHE-BYPASS IS SET AND THE TEST DATA (WHICH HAS BEEN CACHED IN GROUP 0) IS REFERENCED. THE REFERENCES ARE CHECKED FOR MISSES (THE TEST-DATA INSIDE THE CACHE GROUP-0 SHOULD HAVE BEEN INVALIDATED WHEN REFERENCES WERE MADE WITH CACHE-BYPASS SET.) THE ENTIRE TEST IS REPEATED, SELECTING THE OTHER VALID STORE AND THEN WITH TEST-DATA IN GROUP 1.

TEST 52 CHECK CACHE IS BYPASSED ON ASRB OPERAND

THIS TEST CHECKS THAT THE CACHE IS BYPASSED ON THE OPERAND OF THE ASRB INSTRUCTION AND ALSO THE OPERAND IS INVALIDATED. TEST-CODE (INCLUDING THE OPERAND OF THE ASRB) IS MADE HIT IN GROUP 1. THEN ASRB INSTRUCTION IS EXECUTED ON THE CACHED OPERAND. IT IS CHECKED IF THE REFERENCE TO THE BYTE-OPERAND WAS A MISS. THEN THE SAME OPERAND REFERENCED USING AN ORDINARY (NON-BYPASSING) INSTRUCTION. AGAIN, THE REFERENCE IS CHECKED FOR A MISS.

TEST 53 CHECK CACHE VALID STORE PARITY CHECKER

THIS TEST FORCES VALID STORE PARITY ERROR IN THE FOUR VALID STORES AND CHECKS THE PARITY CHECKERS.

TEST 54 CHECK THAT CACHE-MISS OCCURS ON A VALID STORE  
PARITY ERROR <sup>L 2</sup>

THIS TEST FORCES A VALID STORE PARITY ERROR AND CHECKS THAT A MISS OCCURS ON THE REFERENCE THAT CAUSED THE PARITY ERROR. THE CACHE LOCATION THAT GAVE THE PARITY ERROR IS INVALIDATED AND A SLOW CYCLE IS PERFORMED TO THE MAIN MEMORY. THIS TEST IS PERFORMED WITH THE 'DISABLE TRAPS' BIT OF THE CACHE CONTROL REGISTER SET, THUS A PARITY ERROR TRAP WILL NOT OCCUR. THIS IS DONE SO THAT THE HIT-MISS REGISTER CAN BE READ WITHOUT LOSING THE INFORMATION CONTAINED IN IT.

TEST 55 CHECK BYP ON KERNEL PAGE BITS  
THIS TEST IS EXECUTED ONLY ON KB11-E/EM/CM

TEST 56 CHECK BYP ON SUPERVISOR PAGE BITS  
THIS TEST IS EXECUTED ONLY ON KB11-E/EM/CM

TEST 57 CHECK BYP ON USER PAGE BITS  
THIS TEST IS EXECUTED ONLY ON KB11-E/EM/CM

TEST 60 CHECK CACHE BYPASS ON VIRTUAL PAGE  
THIS TEST IS EXECUTED ONLY ON KB11-E/EM/CM

18	OPERATIONAL SWITCH SETTINGS
32	BASIC DEFINITIONS
158	CACHE REGISTER DEFINITIONS
168	CPU REGISTER DEFINITIONS
181	MEMORY MANAGEMENT DEFINITIONS
329	UNIBUS MAP REGISTER DEFINITIONS
462	TRAP CATCHER
471	STARTING ADDRESS(ES)
474	ACT11 HOOKS
486	APT PARAMETER BLOCK
512	COMMON TAGS
598	APT MAILBOX-EABLE
632	ERROR POINTER TABLE
1295	INITIALIZE THE COMMON TAGS
1338	TYPE PROGRAM NAME
1345	GET VALUE FOR SOFTWARE SWITCH REGISTER
1515	T1 PARITY ERROR ABORT
1564	T2 PARITY ERROR TRAP
1608	T3 MEM MGT AND PE TRAP PRIORITY ARBITRATION
1720	T4 UNIBUS PARITY ERROR
1791	T5 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ONES
2059	T6 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ZEROES
2288	T7 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ONES
2494	T10 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ZEROES
2690	T11 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS DUAL ADDRESS TEST
2764	T12 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS DUAL ADDRESS TEST
2883	T13 CACHE ADDRESS MEMORY COMPARATOR TEST
3176	T14 CACHE ADDRESS MEMORY COUNT PATTERN TEST
3390	T15 CACHE ADDRESS MEMORY PARITY LOGIC TEST
3672	T16 CACHE ADDRESS MEMORY DUAL ADDRESS TEST, UPWARD
3969	T17 CACHE ADDRESS MEMORY DUAL ADDRESS TEST, DOWNWARD
4267	T20 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ONES TEST
4405	T21 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ZEROES TEST
4540	T22 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATOB ONES TEST
4702	T23 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATOB ZEROES TEST
4864	T24 CACHE ADDRESS MEMORY POWER UP INVALIDATOR TEST
4946	T25 CACHE DATA MULTIPLEXER, CDMX, TEST
5131	T26 CACHE DATA MEMORY ADDRESS DRIVERS TEST
5249	T27 CACHE DATA MEMORY COUNT PATTERN TEST
5417	T30 CACHE DATA MEMORY PARITY CHECKERS LOW BYTE TEST
5572	T31 CACHE DATA MEMORY PARITY CHECKERS HIGH BYTE TEST
5728	T32 CACHE DATA MEMORY WORST CASE NOISE TEST
5968	T33 CACHE DATA MEMORY CHIP SELECTION LOGIC TEST
6319	T34 CACHE DATA MEMORY BYTE ENABLE LOGIC TEST
6732	T35 CACHE ARBITRATION AND HIGH SPEED I/O TEST
7804	T36 MASS BUS WRITE HIT CYCLE, INVALIDATION TEST
8049	T37 CHECK IVSS, VSIU BITS
8095	T40 CHECK VSIU BIT, WITH IVSS ALREADY SET
8126	T41 CHECK VCIP SETS WHEN CF IS SET
8152	T42 CHECK CACHE FLUSH & VALID STORE SWITCHING
8183	T43 CHECK IVSS INHIBITS SWITCHING OF VALID STORE IN USE
8219	T44 CHECK VALID STORES (A & B) FOR GROUP 0
8338	T45 CHECK VALID STORES (A&B) FOR GROUPS 0 & 1
8422	T46 CHECK VALID STORES (A & B ) FOR GROUP 1
8543	T47 CHECK CACHE TURNS OFF WHEN FLUSH IS DONE WITH IVSS SET
8677	T50 CHECK CACHE TURNS OFF ON A BACK-TO-BACK FLUSH

CEKBD-E 11/70 CACHE #2 MACY11 30A(1052) 13-MAR-80 10:38  
CEKBDE.P11 13-MAR-80 09:59 TABLE OF CONTENTS

SEQ 0026

8712	T51	CHECK CACHE-BYPASS
8828	T52	CHECK CACHE IS BYPASSED ON ASRB OPERAND
8886	T53	CHECK CACHE VALID STORE PARITY CHECKER
9005	T54	CHECK THAT CACHE-MISS OCCURS ON A VALID STORE PARITY ERROR
9084	T55	CHECK BYP ON KERNEL PAGE BITS
9121	T56	CHECK BYP ON SUPERVISOR PAGE BITS
9157	T57	CHECK BYP ON USER PAGE BITS
9193	T60	CHECK CACHE BYPASS ON VIRTUAL PAGE
9353		END OF PASS ROUTINE
9397		SCOPE HANDLER ROUTINE
9465		ERROR HANDLER ROUTINE
9523		APT COMMUNICATIONS ROUTINE
9580		TTY INPUT ROUTINE
9721		SAVE AND RESTORE R0-R5 ROUTINES
9767		TYPE ROUTINE
9847		BINARY TO OCTAL (ASCII) AND TYPE
9925		CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
9993		RANDOM NUMBER GENERATOR ROUTINE
10030		TRAP DECODER
10053		TRAP TABLE
10091		POWER DOWN AND UP ROUTINES
10133		ROUTINE TO SIZE MEMORY
10234		DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
10673		SYSTEM DEVICE SIZER
10904		DEVICE HANDLERS
10965		RP04 DISK HANDLER
11125		RS04 DISK HANDLE
11283		RK05 DISK HANDLER
11469		UNIBUS EXERCISER HANDLER
11595		MASS BUS TESTER HANDLER

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56

000001  
160000  
167400  
000200

```

.TITLE CEKBD-E 11/70 CACHE #2
;*COPYRIGHT (C) 1975, 1980
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*
$TN=1
$SWR=160000      ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
$SWR=167400
$SWRMK=200

.SBTTL OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH          USE
;*      -----          -----
;*      15             HALT ON ERROR
;*      14             LOOP ON TEST
;*      13             INHIBIT ERROR TYPEOUTS
;*      12             EXECUTE THE POWER UP INVALIDATOR TEST
;*      11             INHIBIT ITERATIONS
;*      10             BELL ON ERROR
;*      9              LOOP ON ERROR
;*      8              LOOP ON TEST IN SWR<6:0>
;*      7              SKIP EXECUTION OF TESTS WHICH USE MEMORY MANAGEMENT

.SBTTL BASIC DEFINITIONS
;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
STACK= 1100      ;;FIRST ADDRESS OF THE STACK
KERSTK= STACK   ;;KERNEL STACK
SUPSTK= STACK-200 ;;SUPERVISOR STACK
USESTK= STACK-300 ;;USER STACK
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
PS= 177776      ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774  ;;STACK LIMIT REGISTER
PIRQ= 177772   ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570   ;;HARDWARE SWITCH REGISTER
DDISP= 177570  ;;HARDWARE DISPLAY REGISTER
LKS= 177546    ;;LINE CLOCK (KW11-L) STATUS REGISTER

;*MISCELLANEOUS DEFINITIONS
HT= 11         ;;CODE FOR HORIZONTAL TAB
LF= 12        ;;CODE LINE FEED
CR= 15        ;;CODE CARRIAGE RETURN
CRLF= 200     ;;CODE FOR CARRIAGE RETURN-LINE FEED

;*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0        ;;GENERAL REGISTER
R1= %1        ;;GENERAL REGISTER
R2= %2        ;;GENERAL REGISTER
R3= %3        ;;GENERAL REGISTER
    
```

```

57      000004      R4=      %4          ;;GENERAL REGISTER
58      000005      R5=      %5          ;;GENERAL REGISTER
59      000006      R6=      %6          ;;GENERAL REGISTER
60      000007      R7=      %7          ;;GENERAL REGISTER
61      .EQUIV      R0,R10    ;;GENERAL REGISTER
62      .EQUIV      R1,R11    ;;GENERAL REGISTER
63      .EQUIV      R2,R12    ;;GENERAL REGISTER
64      .EQUIV      R3,R13    ;;GENERAL REGISTER
65      .EQUIV      R4,R14    ;;GENERAL REGISTER
66      .EQUIV      R5,R15    ;;GENERAL REGISTER
67      000006      SP=      %6          ;;STACK POINTER
68      .EQUIV      SP,KSP    ;;KERNEL STACK POINTER
69      .EQUIV      SP,SSP    ;;SUPERVISOR STACK POINTER
70      .EQUIV      SP,USP    ;;USER STACK POINTER
71      000007      PC=      %7          ;;PROGRAM COUNTER
72
73      ;*PRIORITY LEVEL DEFINITIONS
74      000000      PR0=     0          ;;PRIORITY LEVEL 0
75      000040      PR1=    40          ;;PRIORITY LEVEL 1
76      000100      PR2=   100          ;;PRIORITY LEVEL 2
77      000140      PR3=   140          ;;PRIORITY LEVEL 3
78      000200      PR4=   200          ;;PRIORITY LEVEL 4
79      000240      PR5=   240          ;;PRIORITY LEVEL 5
80      000300      PR6=   300          ;;PRIORITY LEVEL 6
81      000340      PR7=   340          ;;PRIORITY LEVEL 7
82
83      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
84      100000      SW15=  100000        SW15=  100000
85      040000      SW14=   40000         SW14=   40000
86      020000      SW13=   20000         SW13=   20000
87      010000      SW12=   10000         SW12=   10000
88      004000      SW11=    4000         SW11=    4000
89      002000      SW10=    2000         SW10=    2000
90      001000      SW09=    1000         SW09=    1000
91      000400      SW08=     400         SW08=     400
92      000200      SW07=    200         SW07=    200
93      000100      SW06=    100         SW06=    100
94      000040      SW05=     40         SW05=     40
95      000020      SW04=     20         SW04=     20
96      000010      SW03=     10         SW03=     10
97      000004      SW02=      4         SW02=      4
98      000002      SW01=      2         SW01=      2
99      000001      SW00=      1         SW00=      1
100     .EQUIV      SW09,SW9
101     .EQUIV      SW08,SW8
102     .EQUIV      SW07,SW7
103     .EQUIV      SW06,SW6
104     .EQUIV      SW05,SW5
105     .EQUIV      SW04,SW4
106     .EQUIV      SW03,SW3
107     .EQUIV      SW02,SW2
108     .EQUIV      SW01,SW1
109     .EQUIV      SW00,SW0
110
111     ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
112     100000      BIT15=  100000
    
```

```

113      040000      BIT14= 40000
114      020000      BIT13= 20000
115      010000      BIT12= 10000
116      004000      BIT11= 4000
117      002000      BIT10= 2000
118      001000      BIT09= 1000
119      000400      BIT08= 400
120      000200      BIT07= 200
121      000100      BIT06= 100
122      000040      BIT05= 40
123      000020      BIT04= 20
124      000010      BIT03= 10
125      000004      BIT02= 4
126      000002      BIT01= 2
127      000001      BIT00= 1
128      .EQUIV BIT09,BIT9
129      .EQUIV BIT08,BIT8
130      .EQUIV BIT07,BIT7
131      .EQUIV BIT06,BIT6
132      .EQUIV BIT05,BIT5
133      .EQUIV BIT04,BIT4
134      .EQUIV BIT03,BIT3
135      .EQUIV BIT02,BIT2
136      .EQUIV BIT01,BIT1
137      .EQUIV BIT00,BIT0
138
139      ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
140      000004      ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
141      000010      RESVEC= 10         ;;RESERVED AND ILLEGAL INSTRUCTIONS
142      000014      TBITVEC=14        ;;'T' BIT
143      000014      TRTVEC= 14         ;;TRACE TRAP
144      000014      BPTVEC= 14        ;;BREAKPOINT TRAP (BPT)
145      000020      IOTVEC= 20         ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
146      000024      PWRVEC= 24         ;;POWER FAIL
147      000030      EMTVEC= 30        ;;EMULATOR TRAP (EMT) **ERROR**
148      000034      TRAPVEC=34        ;;'TRAP' TRAP
149      000060      TKVEC= 60          ;;TTY KEYBOARD VECTOR
150      000064      TPVEC= 64          ;;TTY PRINTER VECTOR
151      000100      LKVEC= 100         ;;LINE CLOCK (KW11-L) VECTOR
152      000114      CACHVEC=114       ;;CACHE ERROR INTERRUPT VECTOR
153      000240      PIRQVEC=240       ;;PROGRAM INTERRUPT REQUEST VECTOR
154      000250      MMVEC= 250        ;;MEMORY MANAGEMENT VECTOR
155      .SBTTL CACHE REGISTER DEFINITIONS
156
157
158      177740      LOADRS = 177740     ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
159      177742      HIADRS = 177742     ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
160      177744      MEMERR = 177744     ;;CACHE ERROR REGISTER
161      177746      CONTRL = 177746    ;;MEMORY CONTROL REGISTER
162      177750      MAINT = 177750     ;;MEMORY MAINTENENCE REGISTER
163      177752      HITMIS = 177752    ;;HIT MISS REGISTER '1' IMPLIES HIT IN CACHE
164
165      .SBTTL CPU REGISTER DEFINITIONS
166
167
168      177760      SIZELO = 177760     ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
    
```

169			:::TO GET TO THE LAST 32 WORDS OF MEMORY
170	177762	SIZEHI = 177762	:::HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
171			:::CURRENTLY ALL ZERO
172	177764	SYSTID = 177764	:::SYSTEM ID REGISTER
173	177766	CPUERR = 177766	:::CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
174			:::THE TRAP TO ERRVEC (000004)

.SBTTL MEMORY MANAGEMENT DEFINITIONS

;\*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

182		MMR0= 177572
183	177572	MMR1= 177574
184	177574	MMR2= 177576
185	177576	MMR3= 172516
186	172516	.EQUIV MMR0,SR0
187		.EQUIV MMR1,SR1
188		.EQUIV MMR2,SR2
189		.EQUIV MMR3,SR3

;\*USER 'I' PAGE DESCRIPTOR REGISTERS

193		UIPDR0= 177600
194	177600	UIPDR1= 177602
195	177602	UIPDR2= 177604
196	177604	UIPDR3= 177606
197	177606	UIPDR4= 177610
198	177610	UIPDR5= 177612
199	177612	UIPDR6= 177614
200	177614	UIPDR7= 177616
201	177616	

;\*USER 'D' PAGE DESCRIPTOR REGISTORS

204		UDPDR0= 177620
205	177620	UDPDR1= 177622
206	177622	UDPDR2= 177624
207	177624	UDPDR3= 177626
208	177626	UDPDR4= 177630
209	177630	UDPDR5= 177632
210	177632	UDPDR6= 177634
211	177634	UDPDR7= 177636
212	177636	

;\*USER 'I' PAGE ADDRESS REGISTERS

215		UIPAR0= 177640
216	177640	UIPAR1= 177642
217	177642	UIPAR2= 177644
218	177644	UIPAR3= 177646
219	177646	UIPAR4= 177650
220	177650	UIPAR5= 177652
221	177652	UIPAR6= 177654
222	177654	UIPAR7= 177656
223	177656	
224		



```

225      ;*USER 'D' PAGE ADDRESS REGISTERS
226
227      177660      UDPAR0= 177660
228      177662      UDPAR1= 177662
229      177664      UDPAR2= 177664
230      177666      UDPAR3= 177666
231      177670      UDPAR4= 177670
232      177672      UDPAR5= 177672
233      177674      UDPAR6= 177674
234      177676      UDPAR7= 177676
235
236      ;*SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS
237
238      172200      SIPDR0= 172200
239      172202      SIPDR1= 172202
240      172204      SIPDR2= 172204
241      172206      SIPDR3= 172206
242      172210      SIPDR4= 172210
243      172212      SIPDR5= 172212
244      172214      SIPDR6= 172214
245      172216      SIPDR7= 172216
246
247      ;*SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS
248
249      172220      SDPDR0= 172220
250      172222      SDPDR1= 172222
251      172224      SDPDR2= 172224
252      172226      SDPDR3= 172226
253      172230      SDPDR4= 172230
254      172232      SDPDR5= 172232
255      172234      SDPDR6= 172234
256      172236      SDPDR7= 172236
257
258      ;*SUPERVISOR 'I' PAGE ADDRESS REGISTERS
259
260      172240      SIPAR0= 172240
261      172242      SIPAR1= 172242
262      172244      SIPAR2= 172244
263      172246      SIPAR3= 172246
264      172250      SIPAR4= 172250
265      172252      SIPAR5= 172252
266      172254      SIPAR6= 172254
267      172256      SIPAR7= 172256
268
269      ;*SUPERVISOR 'D' PAGE ADDRESS REGISTERS
270
271      172260      SDPAR0= 172260
272      172262      SDPAR1= 172262
273      172264      SDPAR2= 172264
274      172266      SDPAR3= 172266
275      172270      SDPAR4= 172270
276      172272      SDPAR5= 172272
277      172274      SDPAR6= 172274
278      172276      SDPAR7= 172276
279
280      ;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS

```

281		KIPDR0= 172300
282	172300	
283	172302	KIPDR1= 172302
284	172304	KIPDR2= 172304
285	172306	KIPDR3= 172306
286	172310	KIPDR4= 172310
287	172312	KIPDR5= 172312
288	172314	KIPDR6= 172314
289	172316	KIPDR7= 172316

;\*KERNEL 'D' PAGE DESCRIPTOR REGISTERS

292		KDPDR0= 172320
293	172320	
294	172322	KDPDR1= 172322
295	172324	KDPDR2= 172324
296	172326	KDPDR3= 172326
297	172330	KDPDR4= 172330
298	172332	KDPDR5= 172332
299	172334	KDPDR6= 172334
300	172336	KDPDR7= 172336

;\*KERNEL 'I' PAGE ADDRESS REGISTERS

304	172340	KIPAR0= 172340
305	172342	KIPAR1= 172342
306	172344	KIPAR2= 172344
307	172346	KIPAR3= 172346
308	172350	KIPAR4= 172350
309	172352	KIPAR5= 172352
310	172354	KIPAR6= 172354
311	172356	KIPAR7= 172356

;\*KERNEL 'D' PAGE ADDRESS REGISTERS

315	172360	KDPAR0= 172360
316	172362	KDPAR1= 172362
317	172364	KDPAR2= 172364
318	172366	KDPAR3= 172366
319	172370	KDPAR4= 172370
320	172372	KDPAR5= 172372
321	172374	KDPAR6= 172374
322	172376	KDPAR7= 172376

.SBTTL UNIBUS MAP REGISTER DEFINITIONS

;\*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'  
;\*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'

332		MAPL00 = 170200
333	170200	
334	170202	MAPH00 = 170202
335	170204	MAPL01 = 170204
336	170206	MAPH01 = 170206

337	170210	MAPL02 = 170210
338	170212	MAPH02 = 170212
339	170214	MAPL03 = 170214
340	170216	MAPH03 = 170216
341	170220	MAPL04 = 170220
342	170222	MAPH04 = 170222
343	170224	MAPL05 = 170224
344	170226	MAPH05 = 170226
345	170230	MAPL06 = 170230
346	170232	MAPH06 = 170232
347	170234	MAPL07 = 170234
348	170236	MAPH07 = 170236
349	170240	MAPL10 = 170240
350	170242	MAPH10 = 170242
351	170244	MAPL11 = 170244
352	170246	MAPH11 = 170246
353	170250	MAPL12 = 170250
354	170252	MAPH12 = 170252
355	170254	MAPL13 = 170254
356	170256	MAPH13 = 170256
357	170260	MAPL14 = 170260
358	170262	MAPH14 = 170262
359	170264	MAPL15 = 170264
360	170266	MAPH15 = 170266
361	170270	MAPL16 = 170270
362	170272	MAPH16 = 170272
363	170274	MAPL17 = 170274
364	170276	MAPH17 = 170276
365	170300	MAPL20 = 170300
366	170302	MAPH20 = 170302
367	170304	MAPL21 = 170304
368	170306	MAPH21 = 170306
369	170310	MAPL22 = 170310
370	170312	MAPH22 = 170312
371	170314	MAPL23 = 170314
372	170316	MAPH23 = 170316
373	170320	MAPL24 = 170320
374	170320	MAPH24 = 170320
375	170324	MAPL25 = 170324
376	170326	MAPH25 = 170326
377	170330	MAPL26 = 170330
378	170332	MAPH26 = 170332
379	170334	MAPL27 = 170334
380	170336	MAPH27 = 170336
381	170340	MAPL30 = 170340
382	170342	MAPH30 = 170342
383	170344	MAPL31 = 170344
384	170346	MAPH31 = 170346
385	170350	MAPL32 = 170350
386	170352	MAPH32 = 170352
387	170354	MAPL33 = 170354
388	170356	MAPH33 = 170356
389	170360	MAPL34 = 170360
390	170362	MAPH34 = 170362
391	170364	MAPL35 = 170364
392	170366	MAPH35 = 170366

393	170370	MAPL36 = 170370
394	170372	MAPH36 = 170372
395	170374	MAPL37 = 170374
396	170376	MAPH37 = 170376
397		.EQUIV MAPL00,MAPL0
398		.EQUIV MAPH00,MAPH0
399		.EQUIV MAPL01,MAPL1
400		.EQUIV MAPH01,MAPH1
401		.EQUIV MAPL02,MAPL2
402		.EQUIV MAPH02,MAPH2
403		.EQUIV MAPL03,MAPL3
404		.EQUIV MAPH03,MAPH3
405		.EQUIV MAPL04,MAPL4
406		.EQUIV MAPH04,MAPH4
407		.EQUIV MAPL05,MAPL5
408		.EQUIV MAPH05,MAPH5
409		.EQUIV MAPL06,MAPL6
410		.EQUIV MAPH06,MAPH6
411		.EQUIV MAPL07,MAPL7
412		.EQUIV MAPH07,MAPH7

:DEFINITIONS

416	100000	VSPE=BIT15
417	040000	IVSS=BIT14
418	020000	VSIU=BIT13
419	010000	VCIP=BIT12
420	004000	DMMA=BIT11
421	002000	FVPE=BIT10
422	001000	UCB=BIT9
423	000400	FCAC=BIT8
424	000040	S1=BIT5
425	000020	S0=BIT4
426	000010	M1=BIT3
427	000004	M0=BIT2
428	000002	DUT=BIT1
429	000001	DT=BIT0
430		
431	100000	BYP=BIT15
432		
433	000054	S1MOM1=BIT5+BIT3+BIT2
434	000034	S0MOM1=BIT4+BIT3+BIT2
435	000014	MOM1=BIT3+BIT2
436		
437	177746	CONTRL=177746
438	177752	HITMIS=177752
439	177744	MSER=177744

440		
441		
442		
443		
444		
445		
446		
447		
448	000011	TAB=11

```

449      000044      S1M0=44
450      000030      SOM1=30
451      000054      S1MOM1=54
452      000034      SOMOM1=34
453      000014      M1M0=14
454      000014      MOM1=M1M0
455      140000      TESTR1=140000
456      142000      TESTR2=142000
457      144000      TESTR3=144000
458      001400      STACK=1400
459      .SBTTL TRAP CATCHER
460
461      000000      .=0
462      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
463      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
464      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
465      .=174
466 000174 000000  DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
467 000176 000000  SWREG:   .WORD 0          ;;SOFTWARE SWITCH REGISTER
468      .SBTTL STARTING ADDRESS(ES)
469 000200 000137 004146  JMP      @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
470
471      .SBTTL ACT11 HOOKS
472
473      ;*****
474      ;HOOKS REQUIRED BY ACT11
475      $SVPC=.          ;SAVE PC
476      .=46
477 000046 051464  $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
478      .=52
479 000052 000000  .WORD 0          ;;2)SET LOC.52 TO ZERO
480      .= $SVPC          ;; RESTORE PC
481
482      .=1400
483      .SBTTL APT PARAMETER BLOCK
484
485      ;*****
486      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
487      ;*****
488      .$X=.          ;;SAVE CURRENT LOCATION
489      .=24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
490 000024 000200  200          ;;FOR APT START UP
491      .=44          ;;POINT TO APT INDIRECT ADDRESS PNTR.
492 000044 001400  $APTHDR      ;;POINT TO APT HEADER BLOCK
493      .= $X          ;;RESET LOCATION COUNTER
494
495      ;*****
496      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
497      ;INTERFACE SPEC.
498 001400  $APTHD:
499 001400 000000  $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
500 001402 001716  $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
501 001404 000010  $STMT:  .WORD 10        ;;RUN TIM OF LONGEST TEST
502 001406 000025  $PASTM: .WORD 25        ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
503 001410 000000  $UNITM: .WORD           ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
504 001412 000014  .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
    
```

505  
506  
507  
508







621			.MEXIT		
622	001746	000	KB11E: .BYTE 0	:1174 WITHOUT MP CACHE FLAG	
623	001747	000	KB11EM: .BYTE 0	:1174 WITH MP CACHE FLAG	
624	001750	000	KB11CM: .BYTE 0	:KB11CM FLAG (1170 WITH MP MODS)	
625	001751	000	CISP: .BYTE 0	:CISP OPTION PRESENT FLAG	
626					
627			:OPCODE FOR MFPT INSTRUCTION (AVAILABLE ON KB11-E AND KB11-EM ONLY)		
628	000007		MFPT=7		

629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684

001752

.SBTTL ERROR POINTER TABLE

;\*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.  
 ;\*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN  
 ;\*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.  
 ;\*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).  
 ;\*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;\* EM ;:POINTS TO THE ERROR MESSAGE  
 ;\* DH ;:POINTS TO THE DATA HEADER  
 ;\* DT ;:POINTS TO THE DATA  
 ;\* DF ;:POINTS TO THE DATA FORMAT

\$ERRTB:

;ERROR TABLE FOR ERROR TYPE OUT:

;ITEM 1 .WORD EM1,DH1,DT1,DF1  
 ;ITEM 2 .WORD EM2,DH2,DT2,DF2  
 ;ITEM 3 .WORD EM3,DH3,DT3,DF3  
 ;ITEM 4 .WORD EM4,DH4,DT4,DF4  
 ;ITEM 5 .WORD EM5,DH5,DT5,DF5  
 ;ITEM 6 .WORD EM6,DH6,DT6,DF6  
 ;ITEM 7 .WORD EM7,DH7,DT7,DF7  
 ;ITEM 10 .WORD EM10,DH10,DT10,DF10  
 ;ITEM 11 .WORD EM11,DH11,DT11,DF11  
 ;ITEM 12 .WORD EM12,DH12,DT12,DF12  
 ;ITEM 13 .WORD EM13,DH13,DT13,DF13  
 ;ITEM 14 .WORD EM14,DH14,DT14,DF14  
 ;ITEM 15 .WORD EM15,DH15,DT15,DF15

072022 111261 115044  
 114521  
 072107 111334 115056  
 114525  
 072301 111334 115056  
 114525  
 072415 111334 115056  
 114525  
 072530 111435 115074  
 114533  
 072610 111435 115074  
 114533  
 072670 111435 115074  
 114533  
 072762 111435 115074  
 114533  
 073053 111467 115116  
 114543  
 073141 111543 115142  
 114554  
 073266 111636 115160  
 114562  
 073346 111711 115174  
 114567  
 073405 112004 115210

685	002120	114574				
686					:ITEM 16	
687	002122	073455	112030	115216	.WORD	EM16,DH16,DT16,DF16
688	002130	114576				
689					:ITEM 17	
690	002132	073531	112116	115234	.WORD	EM17,DH17,DT17,DF17
691	002140	114604				
692					:ITEM 20	
693	002142	073531	112116	115316	.WORD	EM20,DH20,DT20,DF20
694	002150	114604				
695					:ITEM 21	
696	002152	073612	112176	115400	.WORD	EM21,DH21,DT21,DF21
697	002160	114634				
698					:ITEM 22	
699	002162	073676	112251	115456	.WORD	EM22,DH22,DT22,DF22
700	002170	114662				
701					:ITEM 23	
702	002172	074112	112320	115466	.WORD	EM23,DH23,DT23,DF23
703	002200	114665				
704					:ITEM 24	
705	002202	073676	112251	115502	.WORD	EM24,DH24,DT24,DF24
706	002210	114662				
707					:ITEM 25	
708	002212	074112	112320	115512	.WORD	EM25,DH25,DT25,DF25
709	002220	114665				
710					:ITEM 26	
711	002222	074246	112410	115526	.WORD	EM26,DH26,DT26,DF26
712	002230	114672				
713					:ITEM 27	
714	002232	074413	112455	115540	.WORD	EM27,DH27,DT27,DF27
715	002240	114676				
716					:ITEM 30	
717	002242	073676	112251	115554	.WORD	EM30,DH30,DT30,DF30
718	002250	114662				
719					:ITEM 31	
720	002252	074560	112320	115564	.WORD	EM31,DH31,DT31,DF31
721	002260	114665				
722					:ITEM 32	
723	002262	073676	112251	115600	.WORD	EM32,DH32,DT32,DF32
724	002270	114662				
725					:ITEM 33	
726	002272	074560	112320	115610	.WORD	EM33,DH33,DT33,DF33
727	002300	114665				
728					:ITEM 34	
729	002302	074717	112547	115624	.WORD	EM34,DH34,DT34,DF34
730	002310	114703				
731					:ITEM 35	
732	002312	075023	112547	115624	.WORD	EM35,DH35,DT35,DF35
733	002320	114703				
734					:ITEM 36	
735	002322	075132	112627	115640	.WORD	EM36,DH36,DT36,DF36
736	002330	114710				
737					:ITEM 37	
738	002332	075264	112674	115652	.WORD	EM37,DH37,DT37,DF37
739	002340	114714				
740					:ITEM 40	

741	002342	075346	113026	115700	.WORD	EM40,DH40,DT40,DF40
742	002350	114726				
743					:ITEM 41	
744	002352	075521	112761	115666	.WORD	EM41,DH41,DT41,DF41
745	002360	114722				
746					:ITEM 42	
747	002362	075705	112761	115666	.WORD	EM42,DH42,DT42,DF42
748	002370	114722				
749					:ITEM 43	
750	002372	076160	113026	115700	.WORD	EM43,DH43,DT43,DF43
751	002400	114726				
752					:ITEM 44	
753	002402	076306	113075	115722	.WORD	EM44,DH44,DT44,DF44
754	002410	114736				
755					:ITEM 45	
756	002412	076502	113075	115722	.WORD	EM45,DH45,DT45,DF45
757	002420	114736				
758					:ITEM 46	
759	002422	076701	113170	115762	.WORD	EM46,DH46,DT46,DF46
760	002430	114755				
761					:ITEM 47	
762	002432	077022	113170	115762	.WORD	EM47,DH47,DT47,DF47
763	002440	114755				
764					:ITEM 50	
765	002442	076306	113075	116014	.WORD	EM50,DH50,DT50,DF50
766	002450	114736				
767					:ITEM 51	
768	002452	076502	113075	116014	.WORD	EM51,DH51,DT51,DF51
769	002460	114736				
770					:ITEM 52	
771	002462	076701	113170	116054	.WORD	EM52,DH52,DT52,DF52
772	002470	114755				
773					:ITEM 53	
774	002472	077022	113170	116054	.WORD	EM53,DH53,DT53,DF53
775	002500	114755				
776					:ITEM 54	
777	002502	077146	113214	116106	.WORD	EM54,DH54,DT54,DF54
778	002510	114771				
779						
780					:ITEM 55	
781						
782	002512	077321			EM55	
783	002514	113255			DH55	
784	002516	116116			DT55	
785	002520	000000			0	
786						
787						
788						
789					:ITEM 56	
790						
791	002522	077352			EM56	
792	002524	113255			DH55	
793	002526	116116			DT55	
794	002530	000000			0	
795						
796						

797			:ITEM 57
798			
799	002532	077420	EM57
800	002534	113255	DH55
801	002536	116116	DT55
802	002540	114774	DF57
803			
804			
805			:ITEM 60
806			
807	002542	077461	EM60
808	002544	113255	DH55
809	002546	116116	DT55
810	002550	114774	DF61
811			
812			:ITEM 61
813			
814	002552	077520	EM61
815	002554	113255	DH55
816	002556	116116	DT55
817	002560	114774	DF61
818			
819			
820			:ITEM 62
821			
822	002562	077607	EM62
823	002564	113255	DH55
824	002566	116116	DT55
825	002570	114774	DF62
826			
827			
828			:ITEM 63
829			
830	002572	077641	EM63
831	002574	113255	DH55
832	002576	116116	DT55
833	002600	114774	DF63
834			
835			
836			:ITEM 64
837			
838	002602	077717	EM64
839	002604	113255	DH55
840	002606	116116	DT55
841	002610	114774	DF64
842			
843			
844			:ITEM 65
845			
846	002612	100000	EM65
847	002614	113255	DH55
848	002616	116116	DT55
849	002620	114774	DF65
850			
851			
852			:ITEM 66

853			
854	002622	100066	EM66
855	002624	113271	DH66
856	002626	116124	DT66
857	002630	114776	DF66
858			
859			
860			;ITEM 67
861			
862	002632	100176	EM67
863	002634	113271	DH66
864	002636	116124	DT66
865	002640	114776	DF67
866			
867			
868			;ITEM 70
869			
870	002642	100311	EM70
871	002644	113271	DH66
872	002646	116124	DT66
873	002650	114776	DF70
874			
875			;ITEM 71
876			
877	002652	100425	EM71
878	002654	113335	DH71
879	002656	116124	DT66
880	002660	114776	DF71
881			
882			;ITEM 72
883			
884	002662	100473	EM72
885	002664	113271	DH66
886	002666	116124	DT66
887	002670	114776	DF72
888			
889			;ITEM 73
890			
891	002672	100622	EM73
892	002674	113271	DH66
893	002676	116124	DT66
894	002700	114776	DF73
895			
896			;ITEM 74
897			
898	002702	100732	EM74
899	002704	113271	DH66
900	002706	116124	DT66
901	002710	114776	DF74
902			
903			;ITEM 75
904			
905	002712	101034	EM75
906	002714	113271	DH66
907	002716	116124	DT66
908	002720	114776	DF75

909							
910						:ITEM	76
911							
912	002722	101150					EM76
913	002724	113271					DH66
914	002726	116124					DT66
915	002730	114776					DF76
916							
917						:ITEM	77
918							
919	002732	101261					EM77
920	002734	113271					DH66
921	002736	116124					DT66
922	002740	114776					DF77
923							
924						:ITEM	0
925							
926	002742	000000	000000	000000		.WORD	0,0,0,0
927	002750	000000					
928							
929						:ITEM	0
930							
931	002752	000000	000000	000000		.WORD	0,0,0,0
932	002760	000000					
933							
934						:ITEM	0
935							
936	002762	000000	000000	000000		.WORD	0,0,0,0
937	002770	000000					
938							
939						:ITEM	103
940	002772	101524				EM103	:NO PARITY ERROR TRAP ON VALID STORE PARITY ERROR
941	002774	113255				DH55	
942	002776	116116				DT55	
943	003000	114774				DF103	
944							
945						:ITEM	104
946							
947	003002	101605				EM104	:TEST-DATA-REFERENCE GIVING VALID STORE PARITY
948	003004	113255				DH55	:ERROR WAS NOT A MISS
949	003006	116116				DT55	
950	003010	114774				DF104	
951							
952						:ITEM	105
953							
954	003012	101711				EM105	:FVPE DID NOT GET CLEARED AFTER VSPE OCCURED
955	003014	113255				DH55	
956	003016	116116				DT55	
957	003020	114774				DF105	
958							
959						:ITEM	106
960							
961	003022	101765				EM106	:VALID-STORE-PARITY-ERROR BIT DID NOT SET IN CCR ONVSPE
962	003024	113255				DH55	
963	003026	116116				DT55	
964	003030	114774				DF106	

965						
966						
967						
968	003032	102055				
969	003034	113367				
970	003036	116154				
971	003040	115010				
972						
973						
974						
975	003042	102174				
976	003044	113255				
977	003046	116116				
978	003050	114774				
979						
980						
981						
982	003052	102222				
983	003054	113441				
984	003056	116116				
985	003060	114774				
986						
987						
988						
989	003062	102304				
990	003064	113255				
991	003066	116116				
992	003070	114774				
993						
994						
995						
996	003072	102345				
997	003074	113271				
998	003076	116124				
999	003100	114776				
1000						
1001						
1002						
1003	003102	000000	000000	000000		
1004	003110	000000				
1005						
1006						
1007						
1008	003112	102403				
1009	003114	113456				
1010	003116	116116				
1011	003120	114774				
1012						
1013						
1014						
1015	003122	000000	000000	000000		
1016	003130	000000				
1017						
1018						
1019						
1020	003132	000000	000000	000000		



1021	003140	000000					
1022							
1023					:ITEM	120	
1024							
1025	003142	000000	000000	000000		.WORD	0,0,0,0
1026	003150	000000					
1027							
1028					:ITEM	121	
1029							
1030	003152	000000	000000	000000		.WORD	0,0,0,0
1031	003160	000000					
1032							
1033					:ITEM	122	
1034							
1035	003162	000000	000000	000000		.WORD	0,0,0,0
1036	003170	000000					
1037							
1038					:ITEM	123	
1039	003172	102520			EM123		:BYP BIT IN KIPDR COULD NOT BE CLEARED
1040	003174	113475			DH123		: PC KIPDR (KIPDR)
1041	003176	116170			DT123		:\$ERRPC,\$REG0,\$REG1,0
1042	003200	115015			DF123		:0,0,0
1043					:ITEM	124	
1044	003202	102566			EM124		:BYP BIT IN KIPDR COULD NOT BE SET
1045	003204	113475			DH123		
1046	003206	116170			DT123		
1047	003210	115015			DF123		
1048							
1049					:ITEM	125	
1050	003212	102630			EM125		:TEST DATA COULD NOT BE MADE HIT
1051	003214	113524			DH125		: PC CCR PARADR PAR PDR TST-DATA-ADR
1052	003216	116200			DT125		:\$ERRPC,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4,0
1053	003220	115002			DF100		
1054							
1055					:ITEM	126	
1056	003222	102670			EM126		:TEST DATA REFERENCE NOT A MISS
1057							:CACHED DATA WAS NOT FORCED A MISS ON VIRTUAL PAGE BYPASS
1058	003224	113524			DH125		
1059	003226	116200			DT125		
1060	003230	115002			DF100		
1061							
1062					:ITEM	127	
1063	003232	103021			EM127		:TEST DATA REFERENCE NOT A MISS
1064							:CACHED DATA WAS NOT INVALIDATED ON VIRTUAL BYPASS
1065	003234	113524			DH125		
1066	003236	116200			DT125		
1067	003240	115002			DF100		
1068					:ITEM	130	
1069	003242	103150			EM130		:BYP BIT IN SIPDR COULD NOT BE CLEARED
1070	003244	113613			DH130		: PC SIPDR (SIPDR)
1071	003246	116170			DT123		
1072	003250	115015			DF123		
1073							
1074					:ITEM	131	
1075	003252	103216			EM131		:BYP IN SIPDR COULD NOT BE SET
1076	003254	113613			DH130		



1133	003460	115034				
1134					:ITEM 152	
1135	003462	107463	114134	116254	.WORD	EM152,DH152,DT152,DF152
1136	003470	115034				
1137					:ITEM 153	
1138	003472	107463	114203	116254	.WORD	EM153,DH153,DT153,DF153
1139	003500	115034				
1140					:ITEM 154	
1141	003502	107544	114265	116264	.WORD	EM154,DH154,DT154,DF154
1142	003510	115037				
1143					:ITEM 155	
1144	003512	107576	114323	116264	.WORD	EM155,DH155,DT155,DF155
1145	003520	115037				
1146					:ITEM 156	
1147	003522	107630	114361	116264	.WORD	EM156,DH156,DT156,DF156
1148	003530	115037				
1149					:ITEM 0	
1150	003532	000000	000000	000000	.WORD	0,0,0,0
1151	003540	000000				
1152					:ITEM 160	
1153	003542	107675	114417	116254	.WORD	EM160,DH160,DT160,DF160
1154	003550	115037				
1155					:ITEM 161	
1156	003552	107727	114445	116254	.WORD	EM161,DH161,DT161,DF161
1157	003560	115037				
1158					:ITEM 162	
1159						
1160	003562	107775	114475	116276	.WORD	EM162,DH162,DT162,DF55
1161	003570	114774				
1162					:ITEM 163	
1163						
1164	003572	110205	114475	116276	.WORD	EM163,DH162,DT162,DF55
1165	003600	114774				
1166					:ITEM 164	
1167						
1168	003602	110277	114475	116276	.WORD	EM164,DH162,DT162,DF55
1169	003610	114774				
1170					:ITEM 165	
1171						
1172	003612	110356	114475	116276	.WORD	EM165,DH162,DT162,DF55
1173	003620	114774				
1174					:ITEM 166	
1175						
1176	003622	110375	114475	116276	.WORD	EM166,DH162,DT162,DF55
1177	003630	114774				
1178					:ITEM 167	
1179						
1180	003632	110461	114475	116276	.WORD	EM167,DH162,DT162,DF55
1181	003640	114774				
1182					:ITEM 170	
1183						
1184	003642	110565	114475	116276	.WORD	EM170,DH162,DT162,DF55
1185	003650	114774				
1186					:ITEM 171	
1187						
1188	003652	110630	114475	116276	.WORD	EM171,DH162,DT162,DF55

1189	003660	114774			
1190					
1191				:ITEM 172	
1192	003662	110674	114475	116276	.WORD EM172,DH162,DT162,DF55
1193	003670	114774			
1194					
1195				:ITEM 173	
1196	003672	110760	114475	116276	.WORD EM173,DH162,DT162,DF55
1197	003700	114774			
1198					
1199				:ITEM 174	
1200	003702	111146	114475	116276	.WORD EM174,DH162,DT162,DF55
1201	003710	114774			
1202					
1203				:ITEM 175	
1204	003712	111211	114475	116276	.WORD EM175,DH162,DT162,DF55
1205	003720	114774			
1206	003722	000016			RS4REG: .WORD 16
1207	003724	172040			RS4CS1: .WORD 172040
1208	003726	000000			RS4WC: .WORD 0
1209	003730	000000			RS4BA: .WORD 0
1210	003732	000000			RS4DA: .WORD 0
1211	003734	000000			RS4CS2: .WORD 0
1212	003736	000000			RS4DS: .WORD 0
1213	003740	000000			RS4ER: .WORD 0
1214	003742	000000			RS4AS: .WORD 0
1215	003744	000000			RS4LA: .WORD 0
1216	003746	000000			RS4DB: .WORD 0
1217	003750	000000			RS4MR: .WORD 0
1218	003752	000000			RS4DT: .WORD 0
1219	003754	000000			RS4BAE: .WORD 0
1220	003756	000000			RS4CS3: .WORD 0
1221					
1222	003760	000026			RP4REG: .WORD 26
1223	003762	176700			RP4CS1: .WORD 176700
1224	003764	000000			RP4WC: .WORD 0
1225	003766	000000			RP4BA: .WORD 0
1226	003770	000000			RP4DA: .WORD 0
1227	003772	000000			RP4CS2: .WORD 0
1228	003774	000000			RP4DS: .WORD 0
1229	003776	000000			RP4RR1: .WORD 0
1230	004000	000000			RP4AS: .WORD 0
1231	004002	000000			RP4LA: .WORD 0
1232	004004	000000			RP4DB: .WORD 0
1233	004006	000000			RP4MR: .WORD 0
1234	004010	000000			RP4DT: .WORD 0
1235	004012	000000			RP4SN: .WORD 0
1236	004014	000000			RP4OF: .WORD 0
1237	004016	000000			RP4DC: .WORD 0
1238	004020	000000			RP4CCC: .WORD 0
1239	004022	000000			RP4RR2: .WORD 0
1240	004024	000000			RP4RR3: .WORD 0
1241	004026	000000			RP4EC1: .WORD 0
1242	004030	000000			RP4EC2: .WORD 0
1243	004032	000000			RP4BAE: .WORD 0
1244	004034	000000			RP4CS3: .WORD 0

1245					
1246	004036	000014		RH4REG: .WORD	14
1247	004040	160100		RH4CS1: .WORD	160100
1248	004042	000000		RH4WC: .WORD	0
1249	004044	000000		RH4BA: .WORD	0
1250	004046	000000		RH4MR2: .WORD	0
1251	004050	000000		RH4CS2: .WORD	0
1252	004052	000000		RH4ST: .WORD	0
1253	004054	000000		RH4ER: .WORD	0
1254	004056	000000		RH4AS: .WORD	0
1255	004060	000000		RH4DR: .WORD	0
1256	004062	000000		RH4DB: .WORD	0
1257	004064	000000		RH4MR1: .WORD	0
1258	004066	000000		RH4DT: .WORD	0
1259					
1260	004070	000002		RH4REX: .WORD	2
1261	004072	160174		RH4AE: .WORD	160174
1262	004074	000000		RH4CS3: .WORD	0
1263					
1264	004076	000007		RK5REG: .WORD	7
1265	004100	177400		RK5DS: .WORD	177400
1266	004102	000000		RK5ER: .WORD	0
1267	004104	000000		RK5CS1: .WORD	0
1268	004106	000000		RK5WC: .WORD	0
1269	004110	000000		RK5BA: .WORD	0
1270	004112	000000		RK5DA: .WORD	0
1271	004114	000000		RK5DB: .WORD	0
1272					
1273					
1274	004116	000006		UBEREG: .WORD	6
1275	004120	170000		UBEDB: .WORD	170000
1276	004122	000000		UBECC: .WORD	0
1277	004124	000000		UBEBA: .WORD	0
1278	004126	000000		UBECR1: .WORD	0
1279	004130	000000		UBECLR: .WORD	0
1280	004132	000000		UBECR2: .WORD	0
1281					
1282				; THESE ARE THE DEVICE TRAP VECTOR ADDRESSES:	
1283	004134	000204		RS4V: .WORD	204
1284	004136	000254		RP4V: .WORD	254
1285	004140	000774		RH4V: .WORD	774
1286	004142	000220		RK5V: .WORD	220
1287	004144	000510		UBEV: .WORD	510
1288					
1289					
1290					
1291	004146	005037	001502	START: CLR	\$STNM
1292				.SBTTL	INITIALIZE THE COMMON TAGS
1293				::CLEAR	THE COMMON TAGS (\$CMTAG) AREA
1294	004152	012706	001500	MOV	#\$CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
1295	004156	005026		CLR	(R6)+ ;:CLEAR MEMORY LOCATION
1296	004160	022706	001540	CMP	#\$SWR,R6 ;:DONE?
1297	004164	001374		BNE	.-6 ;:LOOP BACK IF NO
1298	004166	012706	001400	MOV	#\$STACK,SP ;:SETUP THE STACK POINTER
1299				::INITIALIZE	A FEW VECTORS
1300	004172	012737	051520 000020	MOV	#\$SCOPE,@#IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE

```

1301 004200 012737 000340 000022      MOV      #340,@#IOTVEC+2  ;;LEVEL 7
1302 004206 012737 052006 000030      MOV      #ERROR,@#EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
1303 004214 012737 000340 000032      MOV      #340,@#EMTVEC+2  ;;LEVEL 7
1304 004222 012737 054374 000034      MOV      #STRAP,@#TRAPVEC  ;;TRAP VECTOR FOR TRAP CALLS
1305 004230 012737 000340 000036      MOV      #340,@#TRAPVEC+2;LEVEL 7
1306 004236 012737 054516 000024      MOV      #SPWRDN,@#PWRVEC  ;;POWER FAILURE VECTOR
1307 004244 012737 000340 000026      MOV      #340,@#PWRVEC+2  ;;LEVEL 7
1308 004252 013737 051414 051406      MOV      SENDCT,$EOPCT    ;;SETUP END-OF-PROGRAM COUNTER
1309 004260 005037 001702                CLR      $TIMES           ;;INITIALIZE NUMBER OF ITERATIONS
1310 004264 005037 001704                CLR      $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1311 004270 112737 000001 001515      MOV      #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1312 004276 012737 004276 001506      MOV      #,$SLPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1313 004304 012737 004304 001510      MOV      #,$SLPERR       ;;SETUP THE ERROR LOOP ADDRESS
1314
1315      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
      ;;EQUAL TO A '-1', SETUP FOR A SOFTWARE SWITCH REGISTER.
1316 004312 013746 000004                MOV      @#ERRVEC,-(SP)   ;;SAVE ERROR VECTOR
1317 004316 012737 004352 000004      MOV      #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1318 004324 012737 177570 001540      MOV      #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1319 004332 012737 177570 001542      MOV      #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1320 004340 022777 177777 175172      CMP      #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1321 004346 001012                BNE      66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1322
1323 004350 000403                BR       65$             ;;AND THE HARDWARE SWR IS NOT = -1
1324 004352 012716 004360 64$:      MOV      #65$,(SP)       ;;BRANCH IF NO TIMEOUT
1325 004356 000002                RTI                      ;;SET UP FOR TRAP RETURN
1326 004360 012737 000176 001540 65$:      MOV      #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1327 004366 012737 000174 001542      MOV      #DISPREG,DISPLAY
1328 004374 012637 000004 66$:      MOV      (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1329
1330 004400 005037 001724                CLR      $PASS           ;;CLEAR PASS COUNT
1331 004404 132737 000200 001737      BITB    #APTSIZE,$ENVM   ;;TEST USER SIZE UNDER APT
1332 004412 001403                BEQ      67$             ;;YES,USE NON-APT SWITCH
1333 004414 012737 001740 001540      MOV      #$$SWREG,SWR   ;;NO,USE APT SWITCH REGISTER
1334 004422
1335 67$:
1336      .SBTTL  TYPE PROGRAM NAME
      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1337 004422 005227 177777                INC      #-1             ;;FIRST TIME?
1338 004426 001046                BNE      68$             ;;BRANCH IF NO
1339 004430 022737 051464 000042      CMP      #SENDAD,@#42   ;;ACT-11?
1340 004436 001442                BEQ      68$             ;;BRANCH IF YES
1341 004440 104401 004506                TYPE    ,69$            ;;TYPE ASCIZ STRING
1342      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1343 004444 005737 000042                TST     @#42             ;;ARE WE RUNNING UNDER XXDP/ACT?
1344 004450 001012                BNE      70$             ;;BRANCH IF YES
1345 004452 123727 001736 000001      CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
1346 004460 001406                BEQ      70$             ;;BRANCH IF YES
1347 004462 023727 001540 000176      CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1348 004470 001005                BNE      71$             ;;BRANCH IF NO
1349 004472 104406                GTSWR                    ;;GET SOFT-SWR SETTINGS
1350 004474 000403                BR       71$
1351 004476 112737 000001 001534 70$:      MOV      #1,$AUTOB      ;;SET AUTO-MODE INDICATOR
1352 004504 71$:
1353 004504 000417                BR       68$             ;;GET OVER THE ASCIZ
1354      ;;69$: .ASCIZ <CRLF>'CEKBD-E 11/70 CACHE #2 '<CRLF>
1355 004544 68$:
1356 004544 005227 177777                INC      #-1             ;;TYPE MESSAGE FIRST PASS ONLY
    
```

```

1357 004550 001043
1358 004552 022737 051464 000042
1359 004560 001437
1360 004562 104401 004570
1361 004566 000434
1362 004570 050200 047522 051107 73$: .ASCIZ <CRLF>'PROGRAMMABLE RP04 DRIVES WILL NOT BE USED BY TEST 35'<CRLF>
1363 004576 046501 040515 046102
1364 004604 020105 050122 032060
1365 004612 042040 044522 042526
1366 004620 020123 044527 046114
1367 004626 047040 052117 041040
1368 004634 020105 051525 042105
1369 004642 041040 020131 042524
1370 004650 052123 031440 100065
1371 004656 000
1372 004660
1373 004660
1374
1375
1376
1377
1378
1379 004660 052737 000200 054752
1380 004666 004737 054670
1381 004672 062737 000037 055270
1382
1383 004700 023737 177760 055270
1384
1385 004706 001420
1386 004710 104401 071411
1387 004714 104401 071546
1388 004720 104401 001713
1389 004724 013746 177760
1390 004730 104403
1391 004732 006
1392 004733 000
1393 004734 104401 071575
1394 004740 013746 055270
1395 004744 104403
1396 004746 006
1397 004747 000
1398 004750
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412

BNE 72$
CMP #SENDAD,@#42 ;SKIP IF ACT
BEQ 72$
TYPE ,73$
BR 72$

.EVEN
72$:
;*****
; SIZE MEMORY AND COMPARE IT WITH THE SYSTEM SIZE REGISTER
;PRINT A WARNING MESSAGE IF THEY DISAGREE.

BIS #BIT07,$KT11
JSR PC,$SIZE
ADD #37,$LSTBK ;ADJUST THE SIZE FOR COMPARISON
;TO SIZE REGISTER
CMP @#SIZELO,$LSTBK ;IS THE ACTUAL SIZE REFLECTED BY THE
;SIZE REGISTER?

BEQ OKSIZ
TYPE ,MS01
TYPE ,MS02
TYPE ,$CRLF
MOV @#SIZELO,-(SP) ;;SAVE @#SIZELO FOR TYPEOUT
TYPOS ;;GO TYPE--OCTAL ASCII
.BYTE 6 ;;TYPE 6 DIGIT(S)
.BYTE 0 ;;SUPPRESS LEADING ZEROS
TYPE ,MS03
MOV $LSTBK,-(SP) ;;SAVE $LSTBK FOR TYPEOUT
TYPOS ;;GO TYPE--OCTAL ASCII
.BYTE 6 ;;TYPE 6 DIGIT(S)
.BYTE 0 ;;SUPPRESS LEADING ZEROS

OKSIZ:
;*** TEST FOR VARIOUS KB11 PROCESSORS ***
;
;THIS ROUTINE POLES THE RESULTS OF ATTEMPTS TO SET TO ONE
;CERTAIN CRITICAL BITS THAT ARE KNOWN TO BE OPERATIVE ON A KB11CM,
;OR KB11EM PROCESSOR. IF TWO OUT OF FOUR OF THE TESTS ARE
;POSITIVE THEN THE KB11CM OR KB11EM FLAG IS SET,IF LESS THAN TWO OF THE
;TESTS ARE POSITIVE THEN THE KB11E FLAG OR NO FLAG IS SET. THE DETERMINATION
;OF WHICH PAIR IS VALID IS BASED ON THE RESULTS OF EXECUTING AN MFPT OPCODE
;(OPCODE 7). IF THIS INSTRUCTION TRAPS THIS IS AN KB11CM OR
;A PLAIN 1170 (KB11-B OR KB11-C). IF THE INSTRUCTION DOES NOT TRAP THEN
;THIS IS A KB11-E OR KB11-EM.
;

```

```

1413 004750 105037 001750          KBTST: CLRB    @#KB11CM      ;RESET THE MP FLAG
1414 004754 005037 001746          CLR     @#KB11E      ;CLEAR KB11E AND KB11EM FLAGS
1415 004760 012737 005216 000010  MOV     #MFPTTR,@#RESVEC ;SET UP TRAP ADDRESS FOR MFPT AT RESERV VECTOR
1416 004766 000007                MFPT                ;EXECUTE MFPT. WILL TRAP ON 1170 (KB11B/C) OR
1417                                ;KB11CM (11/74)
1418 004770 012737 000001 001746  MOV     #1,@#KB11E   ;HERE IF KB11E OR KB11EM. SET FLAG
1419 004776 005037 177750          T1:    CLR     @#MAINT    ;CLEAR THE MAINTENANCE REGISTER
1420 005002 005005                CLR     R5          ;RESET THE TEST COUNTER
1421 005004 012700 177746          MOV     #CONTRL,R0   ;GET THE ADDRESS OF...
1422 005010 012701 177750          MOV     #MAINT,R1    ;CCR,MAINT,AND MAPH00...
1423 005014 012702 170202          MOV     #MAPH00,R2   ;AND PLACE IN R0-R2
1424 005020 052710 040000          BIS     #BIT14,(R0)  ;TRY TO SET IVSS BIT
1425 005024 032710 040000          BIT     #BIT14,(R0)  ;DID IT SET?
1426 005030 001403                BEQ     T2          ;NO,GO TO NEXT TEST
1427 005032 042710 040000          BIC     #BIT14,(R0)  ;CLEAR IT.
1428 005036 005205                INC     R5          ;TEST IS POSITIVE
1429 005040 052711 000001 000001  T2:    BIS     #BIT0,(R1)  ;SET EDMA IN MAINT REGISTER
1430 005044 032711 000001          BIT     #BIT0,(R1)
1431 005050 001410                BEQ     T3
1432 005052 052710 004000          BIS     #BIT11,(R0)  ;TRY TO SET DMMA IN CCR
1433 005056 032710 004000          BIT     #BIT11,(R0)
1434 005062 001403                BEQ     T3
1435 005064 042710 004000          BIC     #BIT11,(R0)
1436 005070 005205                INC     R5
1437 005072 042711 000001 000001  T3:    BIC     #BIT0,(R1)  ;MAKE SURE EDMA IS CLEAR
1438 005076 052737 100000 172300  BIS     #BIT15,KIPDR0 ;TRY TO SET BYP ON A PDR
1439 005104 032737 100000 172300  BIT     #BIT15,KIPDR0
1440 005112 001404                BEQ     T4
1441 005114 042737 100000 172300  BIC     #BIT15,KIPDR0
1442 005122 005205                INC     R5
1443 005124 052712 100000 000000  T4:    BIS     #BIT15,(R2)  ;TRY TO SET BYP ON UNIBUS MAP
1444 005130 032712 100000          BIT     #BIT15,(R2)
1445 005134 001403                BEQ     T.END
1446 005136 042712 100000          BIC     #BIT15,(R2)
1447 005142 005205                INC     R5
1448 005144 022705 000002 000002  T.END: CMP     #2,R5    ;IS THE RESULT OF THE TEST >=2
1449 005150 101021                BHI     2$          ;NO,THIS IT A KB11E OR KB11-B/C (11/70)
1450 005152 005000                CLR     R0
1451 005154 005037 177746          CLR     @#CONTRL    ;CLEAR CACHE CONT. REG. AND
1452 005160 013701 177746          3$:    MOV     @#CONTRL,R1 ;WAIT UNTILL VCIP BIT CLEARS
1453 005164 001402                BEQ     4$          ;OR THE COUNT RUNS OUT
1454 005166 005200                INC     R0
1455 005170 001373                BNE     3$
1456 005172 005737 001746          4$:    TST     @#KB11E    ;IS IS A KB11-E OR KB11-EM?
1457 005176 001404                BEQ     1$          ;BR IF NEITHER. MUST BE KB11CM
1458 005200 012737 000400 001746  MOV     #BIT8,@#KB11E ;SET UPPER BYTE (KB11-EM)
1459 005206 000402                BR     2$          ;DONE
1460 005210 105237 001750          1$:    INCB   @#KB11CM   ;YES, FLAG THIS AS A MODIFIED PROCESSOR
1461 005214 000403                2$:    BR     ENDKB    ;DONE DETERMINING WHICH CPU
1462
1463 005216                MFPTTR:                ;HERE IF MFPT TRAPPED. SEE IF 1170 OR KB11CM
1464 005216 012716 004776          MOV     #T1,(SP)    ;SET UP RETURN ADDRESS FOR RTI
1465 005222 000002                RTI                    ;RETURN
1466 005224
1467 005224 005227 177777          ENDKB: INC     #-1    ;FIRST TIME?
1468 005230 001026                BNE     100$        ;BR IF NO
    
```



```

1469 005232 104401 071605          TYPE      ,MSG1          :<15><12>CPU UNDER TEST FOUND TO BE A
1470 005236 005737 001746          TST       @#KB11E       :IS THIS A KB11-E OR KB11-EM?
1471 005242 001011                    BNE       101$          :BR IF EITHER ONE
1472 005244 105737 001750          TSTB     @#KB11CM      :IS IT A 11/74          (KB11CM)
1473 005250 001003                    BNE       1$           :BR IF IT IS
1474 005252 104401 071655          TYPE      ,MSG3          :KB11-B/C<15><12>
1475 005256 000413                    BR        100$        :SKIP OTHER MESSAGE
1476 005260 104401 071667          1$:      TYPE      ,MSG4          :11/74          (KB11CM)<15><12>
1477 005264 000410                    BR        100$        :SKIP CISP MESSAGE
1478 005266 105737 001746          101$:    TSTB     @#KB11E       :IS IT A KB11-E?
1479 005272 001403                    BEQ      102$          :BR IF NOT. MUST BE KB11-EM
1480 005274 104401 071720          TYPE      ,MSG5          :KB11-E<15><12>
1481 005300 000402                    BR        100$        :SKIP KB11-EM MESSAGE
1482 005302 104401 071644          102$:    TYPE      ,MSG2          :KB11-EM<15><12>
1483 005306
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493 005306 005237 056254          LOOP:    INC       MONF          :INCREMENT THE FLAG WHICH INDICATES
1494 005312 001013                    BNE      TOP           :WHETHER OR NOT THE TOP OF MEMORY
1495
1496 005314 013737 000060 056252          MOV      @#TKVEC,MONTTY :IN THE FIRST 28K HAS BEEN SAVED.
1497
1498 005322 012700 002734          MOV      #^D1500,R0     :SAVE THE INITIAL CONTENTS OF THE TTY
1499 005326 012701 120314          MOV      #BOTTOM+4,R1   :KEYBOARD INTERRUPT VECTOR.
1500 005332 012702 160000          MOV      #160000,R2     :IF NOT THEN SAVE IT.
1501 005336 014221                    1$:      MOV      -(R2),(R1)+ :SAVE IT AT THE BOTTOM OF THIS PROGRAM.
1502 005340 077002                    SOB      R0,1$          :GET THE ADDRESS OF THE END OF THE MONITOR.
1503 005342 012737 000044 177770          TOP:     MOV      #4,@#177770 :SAVE 1500 (DEC) LOCATIONS (WORDS)
1504
1505 005350 012737 056054 000060          MOV      #RESMON,@#TKVEC :SET THE KEYBOARD INTERRUPT VECTOR.
1506 005356 012737 000340 000062          MOV      #340,@#TKVEC+2
1507 005364 005077 174156          CLR      @#TKB          :MAKE SURE THE KEYBOARD BUFFER IS CLEAR.
1508 005370 152777 000100 174146          BISB    #BIT6,@#TKS     :TURN ON INTERRUPT ENABLE FOR THE KEYBOARD.
1509 005376 012737 055412 000004          MOV      #CPSPUR,@#4    :SET UP FOR UNEXPECTED ERRORS.
1510 005404 012737 055440 000114          MOV      #SPUR,@#114
1511
1512
1513
1514
1515
1516
1517
1518 005412 000004
1519 005414 012737 005570 001704          TST1:   SCOPE
1520 005422 012737 000014 177746          MOV      #2$, $ESCAPE   :SETUP ESCAPE ADDRESS
1521 005430 012737 005476 001510          MOV      #14,@#CONTRL  :ENSURE MISSES TO BOTH GROUPS
1522 005436 012737 005570 000114          MOV      #7$, $LPERR    :SETUP ERROR LOOP
1523 005444 012737 005524 000004          MOV      #2$, @#CACHVEC :SETUP CACHE VECTOR
1524 005452 012737 005540 000014          MOV      #3$, @#ERRVEC  :SETUP LOCATION 4
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
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

```

:THIS ROUTINE SAVES THE TOP 1500 (DEC) WORDS OF THE FIRST 28K OF  
 :MEMORY. THESE LOCATIONS SHOULD CONTAIN EITHER THE MONITOR OR THE  
 :LOADER WHICH LOADED THE PROGRAM. NOTE THAT TO RESTORE THIS PART  
 :OF CORE, THAT IS TO RESTORE THE LOADER OR MONITOR, ALL THE USER  
 :MUST DO IS TYPE ^C (CONTROL-C), WHILE THIS PROGRAM IS RUNNING.  
 :THIS WILL AUTOMATICALLY RESTORE THE TOP PART OF MEMORY TO ITS STATE  
 :BEFORE THIS PROGRAM WAS STARTED! AFTER THE MONITOR (OR LOADER) HAS BEEN  
 :RESTORED THIS PROGRAM WILL HALT.

\*\*\*\*\*  
 :\*TEST 1 PARITY ERROR ABORT  
 :\*  
 :\* THIS TEST ENSURES THAT A CACHE PARITY ERROR FLAG CAUSES AN ABORT.  
 :\* THIS IS DONE BY FORCING A PARITY ERROR ON AN EVEN WORD.  
 :\*\*\*\*\*

```

1525 005460 012737 005554 000104      MOV    #5$,@#104      ;SETUP LOCATION 104
1526 005466 012704 170000              MOV    #170000,R4     ;PUT MAINTENANCE DATA IN R4
1527 005472 012702 177750              MOV    #MAINT,R2     ;PUT ADDRESS OF MAIN REG IN R2
1528 005476 012706 001400      7$:   MOV    #STACK,SP    ;INITIALIZE THE SP
1529 005502 000402              BR     1$            ;GO TO NEXT INSTRUCTION
1530              005504              LOC=    ;THIS IS
1531              005504              LOC=-3&LOC          ;USED TO MAKE
1532              005510              LOC=LOC+4           ;1$ FALL ON
1533              005510              .=LOC              ;AND EVEN WORD
1534 005510 000240      1$:   NOP                    ;USED TO MAKE BAD PARITY INSTR ON EVEN WORD
1535 005512 010412              MOV    R4,(R2)       ;SET BITS IN MAINT REG
1536 005514 005701              TST    R1            ;EXECUTE INSTR TO CAUSE PE ABORT
1537              ;FAILURE, NO ABORT
1538 005516 005012              CLR    (R2)          ;CLEAR MAINT REG
1539 005520 000240              NOP
1540 005522 104162              ERROR 162            ;NO PE ABORT
1541              ;FAILURE, ABORTED TO WRONG VECTOR
1542 005524 005012      3$:   CLR    (R2)          ;ENSURE MAINT REG CLEAR
1543 005526 000240              NOP
1544 005530 012737 177777 177744      MOV    #-1,@#MEMERR
1545 005536 104163              ERROR 163            ;ABORTED TO LOCATION 4
1546 005540 005012      4$:   CLR    (R2)          ;ENSURE MAINT REG CLEAR
1547 005542 000240              NOP
1548 005544 012737 177777 177744      MOV    #-1,@#MEMERR
1549 005552 104164              ERROR 164            ;ABORTED TO 14
1550 005554 005012      5$:   CLR    (R2)          ;ENSURE MAINT REG CLEAR
1551 005556 000240              NOP
1552 005560 012737 177777 177744      MOV    #-1,@#MEMERR
1553 005566 104165              ERROR 165            ;ABORTED TO 104
1554              ;TEST OK
1555 005570 005012      2$:   CLR    (R2)          ;ENSURE MAINT REG CLEAR
1556 005572 000240              NOP
1557 005574 012737 177777 177744      MOV    #-1,@#MEMERR ;CLEAR MEMORY ERROR REG
1558 005602 012737 055412 000004      MOV    #CPSPUR,@#ERRVEC ;RESET LOCATION 4
1559              ;CONTINUE
1560              ;*****
1561              ;*TEST 2          PARITY ERROR TRAP
1562              ;*
1563              ;* THIS TEST ENSURES THAT A PARITY TRAP FUNCTIONS PROPERLY.
1564              ;* THIS IS DONE BY MAKING THE ODD WORD HAVE BAD PARITY.
1565              ;* IF THE TRAP DOESN'T OCCUR THEN THE PROBLEM IS ON TMCA.
1566              ;* IF A TRAP OCCURS TO THE WRONG VECTOR THE PROBLEM COULD BE
1567              ;* ON TMCA OR UBCB.
1568              ;*****
1569 005610 000004      TST2:  SCOPE
1570 005612 012737 005716 001704      MOV    #3$, $ESCAPE ;SETUP ESCAPE ADDRESS
1571 005620 012737 005652 001510      MOV    #1$, $LPERR  ;SETUP ERROR LOOP
1572 005626 012737 005702 000004      MOV    #2$, @#ERRVEC ;SETUP THE ERROR VECTOR
1573 005634 012737 005716 000114      MOV    #3$, @#CACHVEC ;SETUP THE CACHE VECTOR
1574 005642 012704 170000              MOV    #170000,R4   ;PUT MAINT DATA IN R4
1575 005646 012702 177750              MOV    #MAINT,R2    ;PUT MAINT REG ADDR IN R2
1576 005652 012706 001400      1$:   MOV    #STACK,SP    ;INITIALIZE THE SP
1577 005656 000402              BR     4$            ;GO TO NEXT INSTRUCTION
1578              005660              LOC=    ;THIS IS USED
1579              005660              LOC=-3&LOC          ;TO MAKE
1580              005664              LOC=LOC+4           ;1$ FALL ON
    
```

```

1581      005664 005664      . =LOC      ; AN EVEN WORD
1582 005664 000240 4$: NOP      ; GOOD PARITY ON EVEN WORD
1583 005666 010412   MOV     R4,(R2) ; SET BITS IN MAINT REG
1584 005670 000240   NOP
1585 005672 005701   TST     R1
1586      005674 005012 ; FAILURE, NO TRAP
1587      005676 000240   CLR     (R2)   ; ENSURE MAINT REG CLEAR
1588      005700 104166   ERROR  166    ; NO PE TRAP
1589      005702 005012 ; FAILURE, TRAPPED TO WRONG VECTOR
1590      005704 000240 2$: CLR     (R2)   ; ENSURE MAINT REG CLEAR
1591      005706 012737 177777 177744   MOV     #-1,@#MEMERR ; CLEAR MEM ERROR REG
1592      005714 104167   ERROR  167    ; PE TRAP, TRAPPED TO
1593      005716 005012 ; TEST OK
1594      005720 000240 3$: CLR     (R2)   ; ENSURE MAINT REG CLEAR
1595      005722 012737 177777 177744   MOV     #-1,@#MEMERR ; CLEAR MEM ERROR REG
1596      005730 012737 055412 000004   MOV     #CPSPUR,@#ERRVEC ; RESTORE LOCATION 4
1597      005736 012737 055440 000250   MOV     #SPUR,@#MMVEC ; RESTORE MEM VEC
1598      005744 000004      ; CONTINUE
1599      005746 012737 001400 172354 *****
1600      005754 112737 000004 172314 *TEST 3      MEM MGT AND PE TRAP PRIORITY ARBITRATION
1601      005762 012704 170000      *
1602      005766 012702 177750      * THIS TEST ENSURES THAT THE ARBITRATION LOGIC WORKS FOR MEMORY
1603      005772 012737 040000 140000      * MANAGEMENT AND PARITY ERROR TRAPS.
1604      006000 012737 006052 000240 *****
1605      006006 012737 000340 000252 TST3: SCOPE
1606      006014 012737 006064 000250 1$: MOV     #1400,@#KIPAR6 ; RESTORE PAR6
1607      006022 012737 006030 001510   MOV     #4,@#KIPDR6   ; SETUP PAGE 6 TO TRAP ON ALL ACCESSES
1608      006030 012706 001400      MOV     #170000,R4    ; PUT MAINT REG DATA IN R4
1609      006034 012737 001001 177572   MOV     #MAINT,R2     ; PUT ADDRESS OF MAINT REG IN R2
1610      006042 000235      ; *****
1611      006044 013737 140000 177772 ; PIR6 DISABLED BY MGMT
1612      006052 005037 177572      MOV     #BIT14,@#140000 ; PUT PIR6 ENABLE BIT IN PAGE 6
1613      006056 005037 177772      MOV     #3$,@#PIRQVEC ; SETUP PIRQ VECTOR
1614      006062 104170      MOV     #PR7,@#MMVEC+2 ; SET UP MMVEC PSW
1615      006064 005037 177572      MOV     #4$,@#MMVEC   ; SETUP MEM MGMT VECTOR
1616      006070 005037 177772      MOV     #5$,$LPERR    ; SETUP ERROR LOOP
1617      006074 012737 006146 000240 5$: MOV     #STACK,SP    ; INITIALIZE THE SP
1618      006076 012737 006146 000240   MOV     #1001,@#MMRO  ; TURN RELOCATION ON
1619      006078 012737 006146 000240   SPL     5             ; SET PROCESSOR AT LEVEL 5
1620      006080 012737 006146 000240   MOV     @#140000,@#PIRQ ; SET PIR6 AND MEM MGT TRAP
1621      006082 012737 006146 000240 ; FAILURE, PRI6 CAME THRU
1622      006084 012737 006146 000240 3$: CLR     @#MMRO      ; TURN RELOCATION OFF
1623      006086 012737 006146 000240   CLR     @#PIRQ       ; CLEAR PIR6
1624      006088 012737 006146 000240   ERROR  170         ; PIR6 CAME IN ON
1625      006090 012737 006146 000240 ; *****
1626      006092 012737 006146 000240 ; PIR3 DISABLED BY MGMT
1627      006094 012737 006146 000240 4$: CLR     @#MMRO      ; TURN RELOCATION OFF
1628      006096 012737 006146 000240   CLR     @#PIRQ       ; CLEAR PIR LEVEL 6
1629      006098 012737 006146 000240   MOV     #6$,@#PIRQVEC ; SETUP PIRQ VECTOR
1630      006100 012737 006146 000240
1631      006102 012737 006146 000240
1632      006104 012737 006146 000240
1633      006106 012737 006146 000240
1634      006108 012737 006146 000240
1635      006110 012737 006146 000240
1636      006112 012737 006146 000240
    
```

```

1637 006102 012737 006160 000250      MOV    #7$, @MMVEC      ;SETUP MEM MGT VECTOR
1638 006110 012737 006124 001510      MOV    #8$, $LPERR     ;SETUP ERROR LOOP
1639 006116 012737 004000 140000      MOV    #BIT11, @#140000 ;PUT PIR3 ENABLE BIT IN PAGE 6
1640 006124 012706 001400      8$:   MOV    #STACK, SP      ;INITIALIZE THE SP
1641 006130 012737 001001 177572      MOV    #1001, @MMRO    ;TURN ON RELOCATION
1642 006136 000232      SPL    2                ;LOWER CPU TO LEVEL 2
1643 006140 013737 140000 177772      MOV    @#140000, @#PIRQ ;SET PIR3 & MGMT
1644      ;FAILURE, PIR3 CAME THRU
1645 006146 005037 177572      6$:   CLR    @MMRO          ;TURN OFF RELOCATION
1646 006152 005037 177772      CLR    @#PIRQ          ;CLEAR PIR3
1647 006156 104171      ERROR  171             ;PIR3 CAME IN ON
1648
1649      ;:*****
1650      ;:STACK LIMIT YELLOW DISABLED BY PARITY ERROR
1651 006160 005037 177572      7$:   CLR    @MMRO          ;TURN RELOCATION OFF
1652 006164 005037 177772      CLR    @#PIRQ          ;CLEAR PIR LEVEL 3
1653 006170 012737 006230 001510      MOV    #9$, $LPERR     ;SETUP ERROR LOOP
1654 006176 012737 006254 000004      MOV    #10$, @#ERRVEC  ;SETUP THE ERROR VECTOR
1655 006204 012737 006254 000114      MOV    #10$, @#CACHVEC ;SETUP CACHEVEC
1656 006212 012737 000240 000116      MOV    #PR5, @#CACHVEC+2 ;PUT PRIORITY 5 IN CACHE VECTOR PSW
1657 006220 012704 170000      MOV    #170000, R4     ;PUT MAINT REG DATA IN R4
1658 006224 012702 177750      MOV    #MAINT, R2     ;PUT ADDRESS OF MAINT ON R2
1659 006230 005037 000370      9$:   CLR    @#370          ;ENSURE LOCATION 370 CLEAR
1660 006234 012706 000376      MOV    #376, SP       ;SETUP THE SP TO YELLOW ZONE
1661 006240      BR     11$            ;GO TO 12$
1662      LOC= .             ;THIS MAKES
1663      LOC=-3&LOC       ;THE NEXT INSTRUCTION
1664      LOC=LOC+4        ;FALL ON
1665      .=LOC            ;AN EVEN WORD
1666 006244 010412      11$:  MOV    R4, (R2)       ;SET MAINT REG
1667 006246 000240      NOP                    ;ODD WORD GOOD PARITY
1668 006250 005216      INC    (SP)           ;CAUSE YEL ZONE (GOOD PARITY)
1669 006252 005701      TST   R1              ;ODD WORD BAD PARITY
1670      ;SHOULD TAKE PE TRAP THEN YEL ZONE TRAP
1671 006254 005012      10$:  CLR    (R2)          ;CLEAR MAINTENANCE REGISTER
1672 006256 000240      NOP
1673 006260 022737 000240 000370      CMP    #PR5, @#370    ;DID CACHVEC PSW GET STACKER?
1674 006266 001403      BEQ   12$             ;BRANCH IF YES
1675 006270 012706 001400      MOV    #STACK, SP    ;RESTORE THE SP
1676 006274 104172      ERROR  172           ;YEL ZONE CAME THRU ON PE TRAP
1677      ;:*****
1678      ;:MEMORY MANAGEMENT TRAP DISABLED BY PARITY TRAP
1679 006276 012737 006344 001510      12$:  MOV    #13$, $LPERR   ;SETUP ERROR LOOP
1680 006304 012737 006374 000250      MOV    #15$, @MMVEC   ;SETUP MEM MGT VECTOR
1681 006312 012737 006374 000114      MOV    #15$, @#CACHVEC ;SETUP CACHVEC
1682 006320 012737 000340 000116      MOV    #PR7, @#CACHVEC+2 ;RESTORE EACH VEC PSW
1683 006326 012704 170000      MOV    #170000, R4    ;PUT MAINT DATA IN R4
1684 006332 012702 177750      MOV    #MAINT, R2    ;PUT ADDRESS OF MAINT REG IN R2
1685 006336 112737 000004 172314      MOV    #4, @#KIPDR6  ;ENSURE PAGE 6 TRAPS
1686 006344 012706 001400      13$:  MOV    #STACK, SP    ;INITIALIZE THE SP
1687 006350 012737 001001 177572      MOV    #1001, @MMRO  ;TURN RELOCATION ON
1688 006356 000402      BR     16$
1689      LOC= .
1690      LOC=-3&LOC
1691      LOC=LOC+4
1692      .=LOC
    
```

```

1693 006364 010412      16$: MOV R4,(R2)      ;SET MAINT REG (PARITY GOOD)
1694 006366 000240      NOP                ;ODD WORD PARITY GOOD
1695 006370 005237 140402 INC @#140402        ;INC HAS GOOD PARITY BUT ADDRESS
1696                                     ;HAS BAD PARITY. CAUSES MM TRAP
1697                                     ;AND PE TRAP
1698                                     ;TEST OK
1699 006374 005012      15$: CLR (R2)        ;CLEAR MAINT REG
1700 006376 000240      NOP                ;
1701 006400 005037 177572 CLR @#MMR0         ;TURN RELOCATION OFF
1702 006404 026627 000002 000340 CMP 2(SP),#PR7    ;DID PE TRAP OCCUR FIRST?
1703 006412 001401      BEQ 14$           ;BRANCH IF YES
1704 006414 104173      ERROR 173        ;MEM MGT TRAP CAME
1705 006416 012737 055412 000004 14$: MOV #CPSPUR,@#ERRVEC ;RESTORE LOCATION 4
1706 006424 012737 055440 000114 MOV #SPUR,@#CACHVEC ;RESTORE LOCATION 114
1707 006432 012737 177777 177744 MOV #-1,@#MEMERR   ;CLEAR MEM ERROR REG
1708 006440 005037 177766 CLR @#CPUERR       ;ENSURE CPUERROR CLEAR
1709                                     ;CONTINUE
    
```

```

:*****
: THE NEXT TEST USES THE MAPPING BOX AND THE CACHE TO
: GENERATE A PARITY ERROR ON THE UNIBUS.
:*****
:TEST 4 UNIBUS PARITY ERROR
:
: THIS TEST MAKES A REFERENCE TO MEMORY THRU THE MAPPING
: BOX THAT WILL CAUSE A PARITY ERROR. IF THE ABORT DOESN'T
: HAPPEN THEN THE PROBLEM IS ON UBCB.
:
: NOTE: MAP REGISTER 0 THRU 2 ARE NOT USED IN CASE THE PROGRAM
: IS RUNNING UNDER ACT11.
:*****
    
```

```

1724 006444 000004      TST4: SCOPE
1725 006446 012737 077406 172314 MOV #77406,@#KIPDR6 ;SETUP PDR6
1726 006454 012737 000060 172516 MOV #60,@#MMR3      ;SETUP MMR3
1727 006462 012706 001400      MOV #STACK,SP      ;INITIALIZE THE SP
1728 006466 012700 170220      MOV #MAPL4,R0      ;GET ADDRESS OF MAP REG 4
1729 006472 012701 000032      MOV #32,R1         ;SETUP SOB COUNT
1730 006476 012737 006510 000004 MOV #5$,@#ERRVEC   ;SETUP ERROR VECTOR
1731 006504 005720      8$: TST (R0)+      ;SEE IF MAP REG IS ENABLED
1732 006506 000420      BR 6$            ;BRANCH IF YES
1733 006510 062700 000002      5$: ADD #2,R0      ;ADJUST R0 TO NEXT REGISTER
1734 006514 077105      SOB R1,8$        ;TEST NEXT REGISTER
1735 006516 012706 001400      7$: MOV #STACK,SP    ;RESTORE THE SP
1736 006522 005737 001724      TST $PASS        ;FIRST PASS?
1737 006526 001105      BNE $EOT         ;BRANCH IF NO
1738 006530 032777 040000 173002 BIT #SW14,@#SWR    ;IS TEST BEING LOOPED ON?
1739 006536 001101      BNE $EOT         ;BRANCH IF YES
1740 006540 104401 071730      TYPE EM724       ;TYPE MESSAGE
1741 006544 000137 006742      JMP $EOT         ;GO TO NEXT TEST
1742 006550 005010      6$: CLR (R0)       ;ENSURE MAP REG HIGH CLEAR
1743 006552 162700 000002      SUB #2,R0        ;GET ADDR OF MAP REG LOW
1744 006556 012710 140000      MOV #140000,(R0) ;PUT ADDR OF PAGE 6 IN MAP REG
1745 006562 072027 000005      ASH #5,R0       ;ADJUST ADDR FOR PAR6
1746 006566 052700 170000      BIS #170000,R0  ;SET UNIBUS ADDR BITS
1747 006572 010037 172354      MOV R0,@#KIPAR6  ;PUT IN PAGE 6 PAR
1748 006576 012737 005701 140000 MOV #5701,@#140000 ;PUT WORD WITH PAD PARITY IN 140000
    
```

```

1749 006604 012704 170000      MOV      #170000,R4      ;PUT MAINT REG DATA IN R4
1750 006610 012702 177750      MOV      #MAINT,R2      ;PUT ADDRESS OF MAINT REG IN R2
1751 006614 012737 006636 001510  MOV      #1$, $LPERR    ;SETUP ERROR LOOP
1752 006622 012737 006700 000000  MOV      #4$, @#0       ;SETUP LOCATION ZERO
1753 006630 012737 006720 000114  MOV      #2$, @#CACHVEC ;SETUP CACH VECTOR
1754 006636 012706 001400      MOV      #STACK,SP     ;INITIALIZE THE SP
1755 006642 052737 000001 177572 1$:      BIS      #BIT0, @#MMRO  ;TURN RELOCATION ON
1756 006650 000401      BR       3$           ;GO TO TEST
1757 006652      LOC=      ;
1758 006650      LOC=-3&LOC
1759 006654      LOC=LOC+4
1760 006654      .=LOC
1761 006654 010412      3$:      MOV      R4,(R2)      ;SET BITS IN MAINT REG
1762 006656 000240      NOP
1763 006660 005037 140000      CLR      @#140000     ;GOOD PARITY ON ODD WORD
1764 006660 005037 140000      CLR      @#140000     ;EXECUTE A DATIP THRU THE
1765 006660 005037 140000      CLR      @#140000     ;MAP THAT CAUSES A PE
1766 006664 005012      ;FAILURE, NO ABORT
1767 006666 000240      CLR      (R2)        ;CLEAR MAINT REG
1768 006670 005037 177572      NOP
1769 006674 104174      CLR      @#MMRO      ;TURN RELOCATION OFF
1770 006676 000410      ERROR   174         ;NO UNIBUS PE ABORT
1771 006676 000410      BR       2$
1772 006700 005012      ;TRAPPED TO WRONG VECTOR
1773 006702 000240      4$:      CLR      (R2)        ;ENSURE MAINT REG CLEAR
1774 006704 005037 177572      NOP
1775 006710 012737 177777 000004  CLR      @#MMRO      ;TURN OFF RELOCATION
1776 006716 104175      MOV      #-1, @#ERRVEC ;CLEAR ERROR REGISTER
1777 006716 104175      ERROR   175         ;TRAPPED TO ZERO
1778 006720 005012      ;TEST OK
1779 006722 000240      2$:      CLR      (R2)        ;ENSURE MAINT REG CLEAR
1780 006724 005037 177572      NOP
1781 006730 012737 177777 177744  CLR      @#MMRO      ;TURN RELOCATION OFF
1782 006736 005037 172516      MOV      #-1, @#MEMERR ;CLEAR ERROR REG
1783 006742 005037 172516      CLR      @#MMR3     ;ENSURE MAP TURNED OFF

```

```

$EOT:
*****
: *TEST 5          CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ONES
: *
: *THIS TEST IS A TEST OF BOTH THE AMX, CPU INPUTS, AND
: *THE CACHE ERROR ADDRESS REGISTER. A SET OF ADDRESSES IS
: *GENERATED AND A MAIN MEMORY ADDRESS AND CONTROL LINE
: *PARITY ERROR IS FORCED AT EACH, THEREBY LOCKING UP
: *THE ADDRESS ON THE OUTPUT OF THE AMX IN THE ERROR
: *ADDRESS REGISTER. THE MANNER IN WHICH THIS IS DONE
: *IS AS FOLLOWS: FIRST THE ADDRESS IS GENERATED;
: *THEN, IF IT IS A VALID ADDRESS (THAT IS, IF IT IS NOT
: *BEYOND THE LIMITS OF MEMORY AS DISPLAYED IN THE
: *SYSTEM SIZE REGISTER), THESE THREE INSTRUCTIONS ARE MOVED
: *TO THAT AREA OF MEMORY:
: *
: *   ONE:  MOV      R1,(R2)
: *   TWO:  CLR      (R2)
: *   THREE: RTS      PC
: *
: *2$ IS THE ADDRESS BEING TESTED. THE INSTRUCTION
: *AT ONE IS GIVEN CONTROL BY A 'JSR PC'. R1 IS MADE
: *TO CONTAIN #2 AND R2 CONTAINS THE ADDRESS OF
: *THE MAINTENANCE REGISTER, SO THAT AFTER THE 'MOV R1,(R2)'

```

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

:\*IS EXECUTED A PARITY ERROR SHOULD OCCUR ON THE  
:\*MAIN MEMORY ADDRESS AND CONTROL LINES WHEN THE  
:\*NEXT INSTRUCTION IS FETCHED.  
:\*THE ADDRESSES USED ARE GENERATED FOLLOWINT THIS PATTERN  
:\*  
200000  
:\*  
200002  
:\*  
200004  
:\*  
200010  
:\*  
200020  
:\*  
200040  
:\*  
200100  
:\*  
200200  
:\*  
200400  
:\*  
ETC. TO:  
240000  
:\*  
300000  
:\*  
400000  
:\*  
400002  
:\*  
400004  
:\*  
400010  
:\*  
ETC. TO:  
500000  
:\*  
600000  
:\*  
1000000  
:\*  
1000002  
:\*  
1000004  
:\*  
ETC.

:\*THE PATTERN CONINUES UNTIL AN ADDRESS IS GENERATED THAT  
:\*IS TOO LARGE.  
:\*MEMORY MANAGEMENT IS SET UP TO FULL 22-BIT MODE, SO  
:\*IF THE USER WANTS TO HAVE THE EXECUTION OF THIS  
:\*TEST DELETED HE CAN SIMPLY BY TURNING ON THE APPORPRIATE  
:\*CONSOLE SWITCH WHICH HAS BEEN DESIGNATED FOR THE  
:\*PURPOSE OF DELETING THE EXECUTION OF TESTS WHICH  
:\*MAKE USER OF MEMORY MANAGEMENT.  
:\*

\*\*\*\*\*

006742 000004  
006744 012737 000020 001702  
000005  
006752 012737 007650 055572  
006760 113737 001502 001632  
006766 012737 055440 000114  
006774 104416  
006776 012700 172340  
007002 012701 077406  
007006 012702 172300  
007012 012703 000010  
007016 010122  
007020 077302

TST5: SCOPE  
MOV #20,\$TIMES ;;DO 20 ITERATIONS  
X=\$TN-1  
MOV #TST6,SKAD ;SET THE SKAD REGISTER  
;IN CASE THE TEST ABORTS.  
MOVB \$STSTM,\$TMP0  
MOV #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS  
;SEE IF THIS TEST SHOULD  
;BE EXECUTED. THE CONDITION  
;TEST IS THE DESIGNATED  
;CONSOLE SWITCH.  
;INITIALIZE THE KERNAL  
;SPACE MEMORY MANAGEMENT  
;REGISTERS  
MMSKIP  
MOV #KIPAR0,R0  
MOV #77406,R1  
MOV #KIPDR0,R2  
MOV #10,R3  
1\$: MOV R1,(R2)+  
SOB R3,1\$

```

1861 007022 005020 CLR (R0)+
1862 007024 012720 000200 MOV #200,(R0)+
1863 007030 012720 000400 MOV #400,(R0)+
1864 007034 012720 000600 MOV #600,(R0)+
1865 007040 012720 001000 MOV #1000,(R0)+
1866 007044 012720 001200 MOV #1200,(R0)+
1867 007050 012720 001400 MOV #1400,(R0)+
1868 007054 012710 177600 MOV #177600,(R0)
1869 007060 012737 000020 172516 MOV #20,@MMR3 ;TURN ON MEMORY MANAGEMENT
1870 007066 012737 000001 177572 MOV #1,@MMR0
1871 007074 104417 SIZE ;DETERMINE FROM THE SYSTEM
1872 ;SIZE REGISTER WHAT THE
1873 ;HIGHEST ADDRESSABLE WORD
1874 ;OF MEMORY IS.
1875 007076 000000 XLOADR: .WORD 0 ;LOW ORDER 16-BITS OF THE
1876 007100 000000 XHIADR: .WORD 0 ;ADDRESS AND HIGH ORDER 6-BITS
1877 007102 042737 000002 007076 BIC #2,XLOADR ;SET THE HIGHEST WORD MINUS TWO
1878 ;IN XLOADR.
1879
1880 007110 012737 000014 177746 MOV #MOM1,@CONTRL ;FORCE MISSES TO BOTH GROUPS.
1881
1882 007116 005037 007636 CLR XADR3 ;INITIALIZE STORAGE
1883 007122 005037 007640 CLR XADR3+2 ;LOCATIONS USED TO GENERATE
1884 007126 005037 007626 CLR XADR1 ;THE SERIES OF TEST ADDRESSES.
1885 007132 012737 000001 007630 MOV #1,XADR1+2
1886
1887 007140 X1:
1888
1889 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
1890 007140 023737 007630 007640 CMP XADR1+2,XADR3+2 ;COMPARE THE HIGH ORDER
1891 007146 001006 BNE 64$ ;PARTS OF XADR1 AND ARG2.
1892 007150 023737 007626 007636 CMP XADR1,XADR3 ;COMPARE THE LOW ORDER
1893
1894 007156 001002 BNE 64$ ;PARTS.
1895
1896
1897
1898 007160 000137 007602 JMP X11 ;THEY WERE EQUAL!
1899
1900 007164 103402 64$: BLO 65$
1901 007166 000137 007176 JMP X2 ;THE FIRST ADDRESS IS LARGER
1902 ;THAN THE SECOND!
1903 007172 000137 007602 65$: JMP X11 ;THE FIRST IS LESS THAN THE
1904 ;SECOND.
1905
1906
1907
1908 007176 X2:
1909 ;DOUBLE PRECISION ADDITION, UNSIGNED
1910 007176 013737 007626 007632 MOV XADR1,XADR2
1911 007204 013737 007630 007634 MOV XADR1+2,XADR2+2
1912 007212 063737 007636 007632 ADD XADR3,XADR2
1913 007220 005537 007634 ADC XADR2+2
1914 007224 063737 007640 007634 ADD XADR3+2,XADR2+2
1915
1916

```



```

1917
1918
1919 007232 X3:
1920
1921 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
1922 007232 023737 007634 007100 CMP XADR2+2,XLOADR+2 ;COMPARE THE HIGH ORDER
1923 007240 001006 BNE 64$ ;PARTS OF XADR2 AND ARG2.
1924 007242 023737 007632 007076 CMP XADR2,XLOADR ;COMPARE THE LOW ORDER
1925
1926 007250 001002 BNE 64$ ;PARTS.
1927
1928
1929
1930 007252 000137 007646 JMP XDONE ;THEY WERE EQUAL!
1931
1932 007256 103402 64$: BLO 65$
1933 007260 000137 007646 JMP XDONE ;THE FIRST ADDRESS IS LARGER
1934 ;THAN THE SECOND!
1935 007264 000137 007270 65$: JMP X4 ;THE FIRST IS LESS THAN THE
1936 ;SECOND.
1937
1938 007270 012737 007270 001510 X4: MOV #X4,$LPERR
1939
1940
1941 ;CONVERT THE 22-BIT ADDRESS IN XADR2 TO VIRTUAL ADDRESS
1942 ;WHICH WILL RELOCATE THROUGH KIPAR6; SET UP KIPAR6;
1943 ;TURN ON MEMORY MANAGEMENT; PUT THE INSTRUCTIONS:
1944 : 1$: MOV R1,(R2)
1945 : 2$: CLR (R2)
1946 : 3$: RTS PC
1947 ;AT THE LOCATION BEING TESTED, WITH 2$=TEST ADDRESS;
1948 ;PUT A PATTERN,000002, IN R1 FOR THE MAINTENANCE
1949 ;REGISTER TO FORCE BAD PARITY ON THE MAIN MEMORY
1950 ;ADDRESS AND CONTROL LINES. PUT THE ADDRESS OF
1951 ;THE CACHE MAINTENANCE REGISTER IN R2. PUT THE
1952 ;ADDRESS, X6, IN LOCATION CACHVEC TO TAKE CARE OF THE
1953 ;WHICH IS BEING FORCED. JSR TO THE ABOVE ROUTINE,
1954 ;SO THAT IF THE PARITY ERROR DOES'NT OCCUR
1955 ;THE 'RTS PC', AT 3$ ABOVE, WILL HANDLE IT.
1956
1957 007276 013703 007632 MOV XADR2,R3
1958 007302 013702 007634 MOV XADR2+2,R2
1959 007306 162703 000002 SUB #2,R3
1960 007312 005602 SBC R2
1961
1962 007314 010300 MOV R3,R0
1963 007316 042700 177701 BIC #177701,R0
1964 007322 062700 140000 ADD #140000,R0
1965 007326 073227 177772 ASHC #-6,R2
1966 007332 010337 172354 MOV R3,@#KIPAR6
1967
1968 007336 012737 000020 172516 MOV #20,@#MMR3 ;TURN ON MEMORY
1969 007344 012737 000001 177572 MOV #1,@#MMR0 ;MANAGEMENT.
1970 ;SET UP THE TEST INSTRUCTIONS.
1971 007352 012710 010112 MOV #010112,(R0) ;010112 = 'MOV R1,(R2)'
1972 007356 012760 005012 000002 MOV #005012,2(R0) ;005012 = 'CLR (R2)'
    
```

```

1973 007364 012760 000207 000004      MOV      #000207,4(R0)      ;000207 = 'RTS PC'
1974                                     ;
1975 007372 012701 000002              MOV      #2,R1              ;SET UP THE REGISTERS
1976 007376 012702 177750              MOV      #MAINT,R2
1977                                     ;
1978 007402 012737 007422 000114      MOV      #X6,@#CACHVEC    ;SET UP THE PARITY ERROR
1979 007410 000240                      NOP                          ;TRAP VECTOR AND GO.
1980 007412 004710                      JSR      PC,(R0)
1981                                     ;
1982 007414                               X5:                          ;NO TRAP OR ABORT OCCURRED!
1983                                     ;MAINTENANCE FUNCTION
1984                                     ;FOR BAD PARITY ON
1985 007414 104022                      1$:      ERROR 22
1986 007416 000137 007534              JMP      X9                  ;THE MAIN MEMORY ADDRESS
1987                                     ;AND CONTROL LINES FAILED
1988 007422                               X6:
1989                                     ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
1990                                     ;COMPARE THE HIGH ORDER
1991 007422 023737 007634 177742      CMP      XADR2+2,LOADRS+2  ;PARTS OF XADR2 AND ARG2.
1992 007430 001006                      BNE     64$
1993 007432 023737 007632 177740      CMP      XADR2,LOADRS     ;COMPARE THE LOW ORDER
1994                                     ;PARTS.
1995 007440 001002                      BNE     64$
1996                                     ;
1997                                     ;
1998                                     ;
1999 007442 000137 007460              JMP      X7                  ;THEY WERE EQUAL!
2000                                     ;
2001 007446 103402                      64$:    BLO      65$
2002 007450 000137 007476              JMP      X8                  ;THE FIRST ADDRESS IS LARGER
2003                                     ;THAN THE SECOND!
2004 007454 000137 007476              65$:    JMP      X8                  ;THE FIRST IS LESS THAN THE
2005                                     ;SECOND.
2006                                     ;
2007                                     ;PARITY ERROR OCCURS.
2008 007460 005726                      X7:    TST      (SP)+          ;RESTORE THE STACK.
2009 007462 022626                      CMP      (SP)+,(SP)+      ;AND CONTINUE SINCE
2010 007464 012737 177777 177744      MOV      #-1,@#MEMERR    ;THE CACHE ERROR ADDRESS
2011 007472 000137 007534              JMP      X9                  ;REGISTER WAS SET CORRECTLY.
2012                                     ;
2013 007476 013737 177744 001634      X8:    MOV      @#MEMERR,$TMP1 ;REPORT VALID TEST
2014                                     ;FAILURE.
2015 007504 013737 177740 001640      MOV      @#LOADRS,$TMP3
2016 007512 013737 177742 001642      MOV      @#HIADRS,$TMP4
2017 007520 005726                      TST      (SP)+
2018 007522 022626                      CMP      (SP)+,(SP)+
2019 007524 104023                      ERROR 23
2020 007526 012737 177777 177744      MOV      #-1,@#MEMERR
2021                                     ;
2022 007534 005037 177572                      X9:    CLR      @#MMR0          ;TURN OFF MEMORY MANAGEMENT.
2023 007540 005037 172516                      CLR      @#MMR3
2024 007544 005737 007636                      TST      XADR3
2025 007550 001007                      BNE     X10
2026 007552 005737 007640                      TST      XADR3+2
2027 007556 001004                      BNE     X10
2028 007560 012737 000002 007636      MOV      #2,XADR3
    
```

```

2029 007566 000415
2030
2031 007570 006337 007636
2032 007574 006137 007640
2033 007600 000410
2034
2035 007602 006337 007626
2036 007606 006137 007630
2037 007612 005037 007636
2038 007616 005037 007640
2039 007622 000137 007140
2040
2041 007626 000000
2042 007630 000000
2043 007632 000000
2044 007634 000000
2045 007636 000000
2046 007640 000000
2047 007642 000000
2048 007644 000000
2049 007646 104414

```

```

BR X12
X10: ASL XADR3
ROL XADR3+2
BR X12
X11: ASL XADR1
ROL XADR1+2
CLR XADR3
CLR XADR3+2
X12: JMP X1

```

```

XADR1: .WORD 0
        .WORD 0
XADR2: .WORD 0
        .WORD 0
XADR3: .WORD 0
        .WORD 0
XADR4: .WORD 0
        .WORD 0
XDONE: RSET

```

;DONE!

```

*****
;*TEST 6          CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ZEROES
;*
;*THIS IS ANOTHER TEST OF THE AMX WHICH IS CARRIED
;*OUT USING THE SAME METHOD AS IN THE PREVIOUS TEST
;*ALL THAT IS DIFFERENT IS THE SERIES OF TEST ADDRESSES
;*WHICH IS USED. IN THE PREVIOUS TEST A ONE WAS
;*FLOATED THROUGH A FIELD OF ZEROES TO PRODUCE THE
;*TEST ADDRESSES, HERE A ZERO WIL BE FLOATED THROUGH
;*A FIELD OF ONES TO PRODUCE THE ADDRESSES
;*BASE ADDRESSES WHICH ARE USE ARE:
;*
;*          177776
;*          377776
;*          777776
;*          1777776
;*          3777776
;*          7777776
;*          17777776
;*EACH OF THESE PATTERNS IS TAKEN AND A ZERO IS FLOATED
;*THROUGHT THE FIELD OF ONES TO PRODUCE A TEST ADDRESS.
;*

```

```

2072
2073 007650 000004
2074 007652 012737 000020 001702
2075 000006
2076
2077 007660 012737 010550 055572
2078
2079 007666 113737 001502 001632
2080 007674 012737 055440 000114
2081
2082 007702 104416
2083
2084

```

```

*****
TST6:  SCOPE
        MOV      #20,$TIMES      ;;DO 20 ITERATIONS
XX=$TN-1
        MOV      #TST7,SKAD      ;SET THE SKAD REGISTER
                                       ;IN CASE THE TEST ABORTS.
        MOVB    $TSTNM,$TMP0
        MOV      #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
MMSKIP
        ;THIS TEST MAKES USE OF
        ;MEMORY MANAGEMENT SO SEE
        ;IF THE USER HAS SET THE

```

```

2085
2086
2087
2088 007704 012700 172340      MOV      #KIPAR0,R0
2089 007710 012701 077406      MOV      #77406,R1
2090 007714 012702 172300      MOV      #KIPDR0,R2
2091 007720 012703 000010      MOV      #10,R3
2092 007724 010122      1$:  MOV      R1,(R2)+
2093 007726 077302      SOB      R3,1$
2094 007730 005020      CLR      (R0)+
2095 007732 012720 000200      MOV      #200,(R0)+
2096 007736 012720 000400      MOV      #400,(R0)+
2097 007742 012720 000600      MOV      #600,(R0)+
2098 007746 012720 001000      MOV      #1000,(R0)+
2099 007752 012720 001200      MOV      #1200,(R0)+
2100 007756 012720 001400      MOV      #1400,(R0)+
2101 007762 012710 177600      MOV      #177600,(R0)
2102 007766 012737 000020 172516      MOV      #20,@MMMR3
2103 007774 012737 000001 177572      MOV      #1,@MMMR0
2104 010002 104417
2105 010004 000000      XXLOA:  .WORD 0
2106 010006 000000      XXHIA:  .WORD 0
2107 010010 042737 000002 010004      BIC      #2,XXLOA
2108
2109
2110
2111
2112 010016 012737 000014 177746      MOV      #MOM1,@CONTRL
2113
2114
2115 010024 012737 177776 010526  XX1:  MOV      #177776,XXADR1
2116 010032 005037 010530      CLR      XXADR1+2
2117 010036 012704 000016      MOV      #16,R4
2118 010042 000410      BR       XX3
2119
2120 010044 005204      XX2:  INC      R4
2121 010046 052737 000001 010526      BIS      #1,XXADR1
2122 010054 006337 010526      ASL      XXADR1
2123 010060 006137 010530      ROL      XXADR1+2
2124
2125 010064 012737 000002 010536  XX3:  MOV      #2,XXMASK
2126 010072 005037 010540      CLR      XXMASK+2
2127
2128 010076 010405      MOV      R4,R5
2129 010100 012737 010106 001510      MOV      #XX4,$LPERR
2130
2131 010106 013737 010526 010532  XX4:  MOV      XXADR1,XXADR2
2132 010114 013737 010530 010534      MOV      XXADR1+2,XXADR2+2
2133 010122 043737 010536 010532      BIC      XXMASK,XXADR2
2134 010130 043737 010540 010534      BIC      XXMASK+2,XXADR2+2
2135
2136
2137
2138 010136 023737 010534 010544      ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2139 010144 001006      CMP      XXADR2+2,XXCNST+2
2140 010146 023737 010532 010542      BNE      64$
2140 010146 023737 010532 010542      CMP      XXADR2,XXCNST

```

```

:SWITCH DESIGNATED AS
:THE DON'T USE MEMORY
:MANAGEMENT SWITCH.
:INITIALIZE THE KERNAL MODE
:MEMORY MANAGEMENT REGISTERS

:TRUN ON MEMORY MANAGEMENT

:GET THE LARGEST MEMORY
:WORD ADDRESS INTO XXLOA
:AND XXHIA.
:GET THE ADDRESS OF THE HIGHEST WORD
:WORD MINUS TWO.

:FROM NOW ON FORCE MISSES
:TO BOTH GROUPS.

:INITIALIZE

:TURN ON THE NEXT BIT
:IN THE FIELD OF ONES.

:INITIALIZE THE MASK
:USED TO CREATE THE ZERO
:IN THE FIELD OF ONES.

:DETERMINE THIS TEST ADDRESS.

:COMPARE THE HIGH ORDER
:PARTS OF XXADR2 AND ARG2.
:COMPARE THE LOW ORDER

```

```

2141
2142 010154 001002          BNE      64$          ;PARTS.
2143
2144
2145
2146 010156 000137 010174    JMP      XX5          ;THEY WERE EQUAL!
2147
2148 010162 103402          64$:    BLO      65$
2149 010164 000137 010174    JMP      XX5          ;THE FIRST ADDRESS IS LARGER
2150                                     ;THAN THE SECOND!
2151 010170 000137 010464    65$:    JMP      XX10   ;THE FIRST IS LESS THAN THE
2152                                     ;SECOND.
2153
2154
2155 010174          XX5:
2156
2157 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2158 010174 023737 010534 010006  CMP     XXADR2+2,XXLOA+2 ;COMPARE THE HIGH ORDER
2159 010202 001006          BNE     64$          ;PARTS OF XXADR2 AND ARG2.
2160 010204 023737 010532 010004  CMP     XXADR2,XXLOA    ;COMPARE THE LOW ORDER
2161
2162 010212 001002          BNE     64$          ;PARTS.
2163
2164
2165
2166 010214 000137 010232    JMP     XX6          ;THEY WERE EQUAL!
2167
2168 010220 103402          64$:    BLO     65$
2169 010222 000137 010464    JMP     XX10        ;THE FIRST ADDRESS IS LARGER
2170                                     ;THAN THE SECOND!
2171 010226 000137 010232    65$:    JMP     XX6          ;THE FIRST IS LESS THAN THE
2172                                     ;SECOND.
2173
2174
2175 010232          XX6:
2176
2177 ;CONVERT THE 22-BIT ADDRESS IN XXADR2 TO VIRTUAL ADDRESS
2178 ;WHICH WILL RELOCATE THROUGH KIPAR6; SET UP KIPAR6;
2179 ;TURN ON MEMORY MANAGEMENT; PUT THE INSTRUCTIONS:
2180 :      1$:    MOV     R1,(R2)
2181 :      2$:    CLR     (R2)
2182 :      3$:    RTS     PC
2183 ;AT THE LOCATION BEING TESTED, WITH 2$=TEST ADDRESS;
2184 ;PUT A PATTERN,000002, IN R1 FOR THE MAINTENANCE
2185 ;REGISTER TO FORCE BAD PARITY ON THE MAIN MEMORY
2186 ;ADDRESS AND CONTROL LINES. PUT THE ADDRESS OF
2187 ;THE CACHE MAINTENANCE REGISTER IN R2. PUT THE
2188 ;ADDRESS, XX7, IN LOCATION CACHVEC TO TAKE CARE OF THE
2189 ;WHICH IS BEING FORCED. JSR TO THE ABOVE ROUTINE,
2190 ;SO THAT IF THE PARITY ERROR DOES'NT OCCUR
2191 ;THE 'RTS PC', AT 3$ ABOVE, WILL HANDLE IT.
2192
2193
2194 010232 013703 010532    MOV     XXADR2,R3
2195 010236 013702 010534    MOV     XXADR2+2,R2
2196 010242 162703 000002    SUB     #2,R3
    
```

D 6

CEKBD-E 11/70 CACHE #2 MACY11 30A(1052) 13-MAR-80 10:38 PAGE 43  
 CEKBDE.P11 13-MAR-80 09:59 T6 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ZEROES SEQ 0068

```

2197 010246 005602          SBC      R2
2198
2199 010250 010300          MOV      R3,R0
2200 010252 042700 177701    BIC      #177701,R0
2201 010256 062700 140000    ADD      #140000,R0
2202 010262 073227 177772    ASHC    #-6,R2
2203 010266 010337 172354    MOV      R3,@#KIPAR6
2204
2205 010272 012737 000020 172516    MOV      #20,@#MMR3      ;TURN ON MEMORY
2206 010300 012737 000001 177572    MOV      #1,@#MMRO      ;MANAGEMENT.
2207                                     ;SET UP THE TEST INSTRUCTIONS.
2208 010306 012710 010112          MOV      #010112,(R0)    ;010112 = 'MOV R1,(R2)'
2209 010312 012760 005012 000002    MOV      #005012,2(R0)  ;005012 = 'CLR (R2)'
2210 010320 012760 000207 000004    MOV      #000207,4(R0)  ;000207 = 'RTS PC'
2211
2212 010326 012701 000002          MOV      #2,R1          ;SET UP THE REGISTERS
2213 010332 012702 177750          MOV      #MAINT,R2
2214
2215 010336 012737 010354 000114    MOV      #XX7,@#CACHVEC ;SET UP THE PARITY ERROR
2216 010344 000240          NOP
2217 010346 004710          JSR      PC,(R0)        ;TRAP VECTOR AND GO.
2218
2219                                     ;NO TRAP OCCURRED!
2220 010350 104024          1$:      ERROR 24
2221 010352 000444          BR       XX10
2222                                     ;COME HERE ON THE PARITY ERROR
2223 010354          XX7:
2224
2225                                     ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2226 010354 023737 010534 177742    CMP      XXADR2+2,LOADRS+2 ;COMPARE THE HIGH ORDER
2227 010362 001006          BNE      64$            ;PARTS OF XXADR2 AND ARG2.
2228 010364 023737 010532 177740    CMP      XXADR2,LOADRS   ;COMPARE THE LOW ORDER
2229
2230 010372 001002          BNE      64$            ;PARTS.
2231
2232
2233
2234 010374 000137 010412          JMP      XX8            ;THEY WERE EQUAL!
2235
2236 010400 103402          64$:     BLO      65$
2237 010402 000137 010426          JMP      XX9            ;THE FIRST ADDRESS IS LARGER
2238                                     ;THAN THE SECOND!
2239 010406 000137 010426          65$:     JMP      XX9            ;THE FIRST IS LESS THAN THE
2240                                     ;SECOND.
2241
2242
2243 010412 005726          XX8:     TST      (SP)+      ;RESTORE THE STACK.
2244 010414 022626          CMP      (SP)+,(SP)+
2245 010416 012737 177777 177744    MOV      #-1,@#MEMERR   ;RESET THE CACHE ERROR REGISTERS.
2246 010424 000417          BR       XX10
2247 010426 013737 177744 001634    XX9:     MOV      @#MEMERR,$TMP1 ;REPORT A VALID TEST
2248                                     ;FAILURE.
2249 010434 013737 177740 001640    MOV      @#LOADRS,$TMP3
2250 010442 013737 177742 001642    MOV      @#HIADRS,$TMP4
2251 010450 005726          TST      (SP)+
2252 010452 022626          CMP      (SP)+,(SP)+

```

```

2253 010454 104025          ERROR 25
2254 010456 012737 177777 177744 MOV    #-1,@#MEMERR
2255
2256 010464 006337 010536    XX10: ASL    XXMASK          ;ROTATE THE MASK.
2257 010470 006137 010540    ROL    XXMASK+2
2258 010474 005305          DEC    R5
2259 010476 001402          BEQ    1$
2260 010500 000137 010106    JMP    XX4
2261 010504 005037 177572    1$:   CLR    @#MMR0          ;TURN OF MEMORY MANAGEMENT.
2262 010510 005037 172516    CLR    @#MMR3
2263 010514 020427 000025    CMP    R4,#25
2264 010520 002012          BGE    XX11
2265 010522 000137 010044    JMP    XX2
2266
2267 010526 000000    XXADR1: .WORD 0          ;USED TO GENERATE TEST PATTERNS.
2268 010530 000000          .WORD 0
2269 010532 000000    XXADR2: .WORD 0          ;USED TO STORE THE CURRENT
2270 010534 000000          .WORD 0          ;TEST PATTERN DURING A TEST.
2271 010536 000000    XXMASK: .WORD 0          ;MASK USED TO PUT A ZERO
2272 010540 000000          .WORD 0          ;IN THE FIELD OF ONES
2273
2274 010542 126310    XXCNST: .WORD BOTPRG     ;TO CREATE A TEST ADDRESS.
2275 010544 000000          .WORD 0          ;THE SMALLEST ADDRESS
2276
2277 010546 104414    XX11:   RSET            ;IN MEMORY OVER THIS TEST.
2278
2279
2280 ::*****
2281 *TEST 7          CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ONES
2282 *
2283 *THIS IS A TEST OF THE UNIBUS INPUTS TO THE AMX.
2284 *THIS TEST IS IDENTICAL TO TST5 IN EVERY THING
2285 *IT DOES EXCEPT IN THAT TEST THE TEST ADDRESSES WERE
2286 *REFERENCED THROUGH MEMORY MANAGEMENT STRAIGHT FROM
2287 *THE CPU TO THE CACHE. HERE THE TEST ADDRESSES WILL
2288 *GO THROUGH THE MEMORY MANAGEMENT UNIT ONTO THE UNIBUS
2289 *WHERE THE MAPPING BOX WILL SEND THEM TO THE CACHE
2290 *AS UNIBUS REFERENCES.
2291 *
2292 ::*****
2293 TST7:  SCOPE
2294 MOV    #20,$TIMES          ;;DO 20 ITERATIONS
2295 RR=$TN-1
2296 MOV    #TST10,SKAD        ;SET THE SKAD REGISTER
2297                                ;IN CASE THE TEST ABORTS.
2298 MOVB   $TSTNM,$TMP0
2299 MOV    #SPUR,@#CACHVEC    ;INITIALLY EXPECT NO ERRORS.
2300 MOV    #CPSPUR,@#ERRVEC
2301
2302 MMSKIP
2303
2304 MOV    #KIPAR0,R0          ;INITIALLY PUT MEMORY
2305 MOV    #77406,R1          ;MANAGEMENT IN A 'PASSIVE'
2306 MOV    #KIPDR0,R2        ;STATE, THAT IS MAP ALL
2307 MOV    #10,R3            ;VIRTUAL ADDRESSES ON TO
2308 64$:  MOV    R1,(R2)+      ;THEMSELVES AS PHYSICAL
    
```

```

2309 010634 077302          SOB      R3,64$          ;ADDRESSES.
2310 010636 005020          CLR      (R0)+
2311 010640 012720 000200    MOV      #200,(R0)+
2312 010644 012720 000400    MOV      #400,(R0)+
2313 010650 012720 000600    MOV      #600,(R0)+
2314 010654 012720 001000    MOV      #1000,(R0)+
2315 010660 012720 001200    MOV      #1200,(R0)+
2316 010664 012720 001400    MOV      #1400,(R0)+
2317 010670 012710 177600    MOV      #177600,(R0)
2318
2319 010674 012737 000060 172516    MOV      #60,@#MMR3      ;TURN ON MEMORY MANAGEMENT.
2320 010702 012737 000001 177572    MOV      #1,@#MMR0
2321
2322 010710 104417          SIZE          ;DETERMINE THE MEMORY
2323 010712 000000          RRLOAD: .WORD 0        ;SYSTEM SIZE.
2324 010714 000000          RRHIAD: .WORD 0        ;LOW ORDER 16-BITS AND
2325
2326 010716 042737 000002 010712    BIC      #2,RRLOAD      ;HIGH ORDER 6-BITS OF THE
2327
2328 010724 012737 000014 177746    MOV      #MOM1,@#CONTRL ;HIGHEST MEMORY WORD ADDRESS.
2329
2330 010732 005037 011454          CLR      RRADR3          ;INITIALIZE STORAGE LOCATIONS
2331 010736 005037 011456          CLR      RRADR3+2        ;USED TO GENERATE THE
2332 010742 005037 011444          CLR      RRADR1          ;SERIES OF TEST ADDRESSES.
2333 010746 012737 000001 011446    MOV      #1,RRADR1+2
2334
2335 010754          RR1:
2336
2337          ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2338 010754 023737 011446 011456    CMP      RRADR1+2,RRADR3+2 ;COMPARE THE HIGH ORDER
2339 010762 001006          BNE      64$            ;PARTS OF RRADR1 AND ARG2.
2340 010764 023737 011444 011454    CMP      RRADR1,RRADR3    ;COMPARE THE LOW ORDER
2341
2342 010772 001002          BNE      64$            ;PARTS.
2343
2344
2345
2346 010774 000137 011420          JMP      RR11           ;THEY WERE EQUAL!
2347
2348 011000 103402          64$: BLO      65$
2349 011002 000137 011012          JMP      RR2            ;THE FIRST ADDRESS IS LARGER
2350
2351 011006 000137 011420          65$: JMP      RR11      ;THAN THE SECOND!
2352
2353
2354
2355 011012          RR2:
2356          ;DOUBLE PRECISION ADDITION, UNSIGNED
2357 011012 013737 011444 011450    MOV      RRADR1,RRADR2
2358 011020 013737 011446 011452    MOV      RRADR1+2,RRADR2+2
2359 011026 063737 011454 011450    ADD      RRADR3,RRADR2
2360 011034 005537 011452          ADC      RRADR2+2
2361 011040 063737 011456 011452    ADD      RRADR3+2,RRADR2+2
2362
2363
2364
    
```



```

2365
2366 011046 RR3:
2367
2368 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2369 011046 023737 011452 010714 CMP RRADR2+2,RRLOAD+2 ;COMPARE THE HIGH ORDER
2370 011054 001006 BNE 64$ ;PARTS OF RRADR2 AND ARG2.
2371 011056 023737 011450 010712 CMP RRADR2,RRLOAD ;COMPARE THE LOW ORDER
2372
2373 011064 001002 BNE 64$ ;PARTS.
2374
2375
2376
2377 011066 000137 011460 JMP RRDONE ;THEY WERE EQUAL!
2378
2379 011072 103402 64$: BLO 65$
2380 011074 000137 011460 JMP RRDONE ;THE FIRST ADDRESS IS LARGER
2381 ;THAN THE SECOND!
2382 011100 000137 011104 65$: JMP RR4 ;THE FIRST IS LESS THAN THE
2383 ;SECOND.
2384
2385 011104 012737 011104 001510 RR4: MOV #RR4,$LPERR
2386 ;CONVERT THE PHYSICAL 22-BIT, ADDRESS IN RRADR2 TO A VIRTUAL ADDRESS
2387 ;WHICH WILL RELOCATE THROUGH KIPAR6 TO THE UNIBUS, THEN THROUGH
2388 ;THE MAPPING BOX TO THE UNIBUS INPUTS OF THE CACHE AMX.
2389 ;NOTE: MAP REGISTERS 0-2 ARE NOT USED IN CASE PROGRAM IS
2390 ;RUNNING UNDER APT OR ACT.
2391 011112 013737 011450 170214 MOV RRADR2,@#MAPL03 ;SET UP THE MAP REGISTER 3.
2392 011120 013737 011452 170216 MOV RRADR2+2,@#MAPH03
2393 011126 162737 000002 170214 SUB #2,@#MAPL03
2394 011134 005637 170216 SBC @#MAPH03
2395
2396 011140 012700 140000 MOV #140000,R0 ;A VIRTUAL ADDRESS WHICH WILL
2397 ;RELOCATE THROUGH KIPAR6.
2398 011144 012737 170600 172354 MOV #170600,@#KIPAR6 ;RELOCATE TO UNIBUS BASE
2399 ;ADDRESS OF 000000.
2400 011152 012737 000060 172516 MOV #60,@#MMR3 ;TURN ON THE MAPPING BOX AND
2401 ;22-BIT MODE.
2402 011160 012737 000001 177572 MOV #1,@#MMR0 ;TURN ON MEMORY MANAGEMENT.
2403 ;SET UP THE TEST CODE:
2404 011166 012710 010112 MOV #010112,(R0) ;010112='MOV R1,(R2)'
2405 011172 012760 005012 000002 MOV #005012,2(R0) ;005012='CLR (R2)'
2406 011200 012760 000207 000004 MOV #000207,4(R0) ;000207='RTS PC'
2407
2408 011206 012701 000002 MOV #2,R1 ;SET UP THE REGISTERS USED
2409 011212 012702 177750 MOV #MAINT,R2 ;IN THE TEST INSTRUCTIONS.
2410
2411 011216 012737 011236 000114 MOV #RR6,@#CACHVEC ;SET UP THE PARITY TRAP
2412 011224 000240 NOP ;VECTOR.
2413 011226 004710 JSR PC,(R0) ;AND GO.
2414
2415
2416 011230 RR5: ;NO TRAP OR ABORT OCCURRED!
2417 ;MAINTENANCE FUNCTION FOR
2418 011230 104030 1$: ERROR 30 ;FORCING BAD PARITY ON
2419 011232 000137 011352 JMP RR9 ;THE MAIN MEMORY ADDRESS
2420 ;AND CONTROL LINES FAILED.
    
```

```

2421 ;COME HERE WHEN THE FORCED ERROR OCCURS.
2422 011236 RR6:
2423
2424 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2425 011236 023737 011452 177742 CMP RRADR2+2,LOADRS+2 ;COMPARE THE HIGH ORDER
2426 011244 001006 BNE 64$ ;PARTS OF RRADR2 AND ARG2.
2427 011246 023737 011450 177740 CMP RRADR2,LOADRS ;COMPARE THE LOW ORDER
2428
2429 011254 001002 BNE 64$ ;PARTS.
2430
2431
2432
2433 011256 000137 011274 JMP RR7 ;THEY WERE EQUAL!
2434
2435 011262 103402 64$: BLO 65$
2436 011264 000137 011314 JMP RR8 ;THE FIRST ADDRESS IS LARGER
2437 ;THAN THE SECOND!
2438 011270 000137 011314 65$: JMP RR8 ;THE FIRST IS LESS THAN THE
2439 ;SECOND.
2440
2441
2442 011274 022626 RR7: CMP (SP)+,(SP)+
2443 011276 005726 TST (SP)+ ;RESTORE THE STACK.
2444 011300 022626 CMP (SP)+,(SP)+
2445 011302 012737 177777 177744 MOV #-1,@MEMERR ;CLEAR THE CACHE ERROR REGISTER.
2446 011310 000137 011352 JMP RR9
2447
2448 011314 013737 177744 001634 RR8: MOV @MEMERR,$TMP1 ;REPORT A VALID TEST FAILURE.
2449 011322 013737 177740 001640 MOV @LOADRS,$TMP3
2450 011330 013737 177742 001642 MOV @HIADRS,$TMP4
2451 011336 005726 TST (SP)+
2452 011340 022626 CMP (SP)+,(SP)+
2453 011342 104031 ERROR 31
2454 011344 012737 000001 177744 MOV #1,@MEMERR ;CLEAR THE ERROR REGISTER.
2455 011352 005037 177572 RR9: CLR @MMR0 ;TURN OFF MEMORY MANAGEMENT.
2456 011356 005037 172516 CLR @MMR3
2457 011362 005737 011454 TST RRADR3 ;GET READY TO GENERATE THE
2458 011366 001007 BNE RR10 ;NEXT ADDRESS TO BE TESTED.
2459 011370 005737 011454 TST RRADR3
2460 011374 001004 BNE RR10
2461 011376 012737 000002 011454 MOV #2,RRADR3
2462 011404 000415 BR RR12
2463
2464 011406 006337 011454 RR10: ASL RRADR3
2465 011412 006137 011456 ROL RRADR3+2
2466 011416 000410 BR RR12
2467
2468 011420 006337 011444 RR11: ASL RRADR1
2469 011424 006137 011446 ROL RRADR1+2
2470 011430 005037 011454 CLR RRADR3
2471 011434 005037 011456 CLR RRADR3+2
2472
2473 011440 000137 010754 RR12: JMP RR1
2474
2475 011444 000000 RRADR1: .WORD 0 ;3 DOUBLE WORD LOCATIONS
2476 011446 000000 .WORD 0 ;USED TO STORE 22-BIT

```

2477 011450 000000  
 2478 011452 000000  
 2479 011454 000000  
 2480 011456 000000  
 2481  
 2482 011460 104414  
 2483  
 2484  
 2485  
 2486  
 2487  
 2488  
 2489  
 2490  
 2491  
 2492  
 2493  
 2494  
 2495  
 2496  
 2497 011462 000004  
 2498 011464 012737 000020 001702  
 2499 000010  
 2500  
 2501 011472 012737 012350 055572  
 2502  
 2503 011500 113737 001502 001632  
 2504 011506 012737 055440 000114  
 2505 011514 104416  
 2506  
 2507 011516 012700 172340  
 2508 011522 012701 077406  
 2509 011526 012702 172300  
 2510 011532 012703 000010  
 2511 011536 010122  
 2512 011540 077302  
 2513 011542 005020  
 2514 011544 012720 000200  
 2515 011550 012720 000400  
 2516 011554 012720 000600  
 2517 011560 012720 001000  
 2518 011564 012720 001200  
 2519 011570 012720 001400  
 2520 011574 012710 177600  
 2521  
 2522 011600 104417  
 2523 011602 000000  
 2524 011604 000000  
 2525 011606 042737 000002 011602  
 2526  
 2527 011614 012737 000014 177746  
 2528  
 2529 011622 012737 177776 012326  
 2530 011630 005037 012330  
 2531 011634 012704 000016  
 2532 011640 000410

```

RRADR2: .WORD 0 ;ADDRESSES.
        .WORD 0
RRADR3: .WORD 0
        .WORD 0

RRDONE: RSET ;DONE!

:*****
:*TEST 10 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ZEROES
:*
:*THIS IS A TEST OF THE UNIBUS INPUTS TO THE AMX.
:*THIS TEST IS IDENTICAL TO TST6 IN EVERY THING
:*IT DOES EXCEPT IN THAT TEST THE TEST ADDRESSES WERE
:*REFERENCED THROUGH MEMORY MANAGEMENT STRAIGHT FROM
:*THE CPU TO THE CACHE. HERE THE TEST ADDRESSES WILL
:*GO THROUGH THE MEMORY MANAGEMENT UNIT ONTO THE UNIBUS
:*WHERE THE MAPPING BOX WILL SEND THEM TO THE CACHE
:*AS UNIBUS REFERENCES.
:*
:*****
TST10:  SCOPE
        MOV #20,$TIMES ;:DO 20 ITERATIONS
SS=$TN-1
        ;SET THE SKAD REGISTER
        MOV #TST11,SKAD ;:IN CASE THE TEST ABORTS.
        ;INITIALLY EXPECT NO ERRORS
        MOV $TSTNM,$TMP0
        MOV #SPUR,@#CACHVEC
        MMSKIP
        MOV #KIPAR0,R0 ;:INITIALLY PUT MEMORY
        MOV #77406,R1 ;:MANAGEMENT IN A 'PASSIVE'
        MOV #KIPDR0,R2 ;:STATE, THAT IS MAP ALL
        MOV #10,R3 ;:VIRTUAL ADDRESSES ON TO
64$:    MOV R1,(R2)+ ;:THEMSELVES AS PHYSICAL
        SOB R3,64$ ;:ADDRESSES.
        CLR (R0)+
        MOV #200,(R0)+
        MOV #400,(R0)+
        MOV #600,(R0)+
        MOV #1000,(R0)+
        MOV #1200,(R0)+
        MOV #1400,(R0)+
        MOV #177600,(R0)

        ;GET THE MEMORY SIZE.
        ;22-BIT ADDRESS OF THE
        ;HIGHEST WORD IN MEMORY.
        ;GET THE HIGHEST WORD MINUS TWO.
        SIZE
        SSLOAD: .WORD 0
        SSHIAD: .WORD 0
        BIC #2,SSLOAD

        ;INITIALIZE
        MOV #177776,SSADR1
        CLR SSADR1+2
        MOV #16,R4
        BR SS3
  
```

```

2533
2534 011642 005204          SS2:  INC  R4          ;TURN ON THE NEXT BIT
2535 011644 052737 000001 012326  BIS  #1,SSADR1      ;IN THE FIELD OF ONES
2536 011652 006337 012326          ASL  SSADR1
2537 011656 006137 012330          ROL  SSADR1+2
2538
2539 011662 012737 000002 012336  SS3:  MOV  #2,SSMASK      ;INITIALIZE THE MASK USER
2540 011670 005037 012340          CLR  SSMASK+2      ;TO CREATE THE ZERO IN
2541                                     ;IN FIELD OF ONES
2542 011674 010405          MOV  R4,R5
2543 011676 012737 011704 001510  MOV  #SS4,$LPERR
2544
2545 011704 013737 012326 012332  SS4:  MOV  SSADR1,SSADR2      ;DETERMINE THE TEST ADDRESS.
2546 011712 013737 012330 012334  MOV  SSADR1+2,SSADR2+2
2547 011720 043737 012336 012332  BIC  SSMASK,SSADR2
2548 011726 043737 012340 012334  BIC  SSMASK+2,SSADR2+2
2549
2550                                     ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2551 011734 023737 012334 012344  CMP  SSADR2+2,SSCNST+2      ;COMPARE THE HIGH ORDER
2552 011742 001006          BNE  64$              ;PARTS OF SSADR2 AND ARG2.
2553 011744 023737 012332 012342  CMP  SSADR2,SSCNST        ;COMPARE THE LOW ORDER
2554
2555 011752 001002          BNE  64$              ;PARTS.
2556
2557
2558
2559 011754 000137 011772          JMP  SS5              ;THEY WERE EQUAL!
2560
2561 011760 103402          64$: BLO  65$
2562 011762 000137 011772          JMP  SS5              ;THE FIRST ADDRESS IS LARGER
2563                                     ;THAN THE SECOND!
2564 011766 000137 012264          65$: JMP  SS10         ;THE FIRST IS LESS THAN THE
2565                                     ;SECOND.
2566
2567 011772          SS5:
2568
2569                                     ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2570 011772 023737 012334 011604  CMP  SSADR2+2,SSLOAD+2      ;COMPARE THE HIGH ORDER
2571 012000 001006          BNE  64$              ;PARTS OF SSADR2 AND ARG2.
2572 012002 023737 012332 011602  CMP  SSADR2,SSLOAD        ;COMPARE THE LOW ORDER
2573
2574 012010 001002          BNE  64$              ;PARTS.
2575
2576
2577
2578 012012 000137 012030          JMP  SS6              ;THEY WERE EQUAL!
2579
2580 012016 103402          64$: BLO  65$
2581 012020 000137 012264          JMP  SS10             ;THE FIRST ADDRESS IS LARGER
2582                                     ;THAN THE SECOND!
2583 012024 000137 012030          65$: JMP  SS6         ;THE FIRST IS LESS THAN THE
2584                                     ;SECOND.
2585
2586
2587 012030          SS6:
2588                                     ;CONVERT THE PHYSICAL 22-BIT, ADDRESS IN SSADR2 TO A VIRTUAL ADDRESS
    
```

```

2589
2590
2591
2592
2593 012030 013737 012332 170214
2594 012036 013737 012334 170216
2595 012044 162737 000002 170214
2596 012052 005637 170216
2597
2598 012056 012700 140000
2599
2600 012062 012737 170600 172354
2601
2602 012070 012737 000060 172516
2603
2604 012076 012737 000001 177572
2605
2606 012104 012710 010112
2607 012110 012760 005012 000002
2608 012116 012760 000207 000004
2609
2610 012124 012701 000002
2611 012130 012702 177750
2612
2613 012134 012737 012210 000114
2614 012142 000240
2615 012144 004710
2616
2617
2618 012146 104032
2619 012150 000445
2620
2621 012152
2622
2623
2624 012152 023737 012334 177742
2625 012160 001006
2626 012162 023737 012332 177740
2627
2628 012170 001002
2629
2630
2631
2632 012172 000137 012210
2633
2634 012176 103402
2635 012200 000137 012226
2636
2637 012204 000137 012226
2638
2639
2640
2641 012210 022626
2642 012212 005726
2643 012214 022626
2644 012216 012737 177777 177744

```

```

;WHICH WILL RELOCATE THROUGH KIPAR6 TO THE UNIBUS, THEN THROUGH
;THE MAPPING BOX TO THE UNIBUS INPUTS OF THE CACHE AMX.
;NOTE: MAP REGISTERS 0-2 ARE NOT USED IN CASE PROGRAM IS
;RUNNING UNDER APT OR ACT.
MOV SSADR2,@#MAPL03 ;SET UP THE MAP REGISTER 3.
MOV SSADR2+2,@#MAPH03
SUB #2,@#MAPL03
SBC @#MAPH03
MOV #140000,R0 ;A VIRTUAL ADDRESS WHICH WILL
;RELOCATE THROUGH KIPAR6.
MOV #170600,@#KIPAR6 ;RELOCATE TO UNIBUS BASE
;ADDRESS OF 000000.
MOV #60,@#MMR3 ;TURN ON THE MAPPING BOX AND
;22-BIT MODE.
MOV #1,@#MMR0 ;TURN ON MEMORY MANAGEMENT.
MOV #010112,(R0) ;SET UP THE TEST CODE:
;010112='MOV R1,(R2)'
MOV #005012,2(R0) ;005012='CLR (R2)'
MOV #000207,4(R0) ;000207='RTS PC'
MOV #2,R1 ;SET UP THE REGISTERS USED
MOV #MAINT,R2 ;IN THE TEST INSTRUCTIONS.
MOV #SS8,@#CACHVEC ;SET UP THE PARITY TRAP
NOP ;VECTOR.
JSR PC,(R0) ;AND GO.
;NO TRAP OCCURRED!
1$: ERROR 32
RR SS10
;TRAP TO HERE WHEN THE ERROR OCCURS.
SS7:
;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
CMP SSADR2+2,LOADRS+2 ;COMPARE THE HIGH ORDER
BNE 64$ ;PARTS OF SSADR2 AND ARG2.
CMP SSADR2,LOADRS ;COMPARE THE LOW ORDER
BNE 64$ ;PARTS.
JMP SS8 ;THEY WERE EQUAL!
64$: BLO 65$
JMP SS9 ;THE FIRST ADDRESS IS LARGER
;THAN THE SECOND!
65$: JMP SS9 ;THE FIRST IS LESS THAN THE
;SECOND.
SS8: CMP (SP)+,(SP)+
TST (SP)+ ;RESTORE THE STACK
CMP (SP)+,(SP)+
MOV #-1,@#MEMERR ;CLEAR THE CACHE ERROR

```

```

2645 012224 000417 BR SS10 ;REGISTER.
2646
2647 012226 013737 177744 001634 SS9: MOV @#MEMERR,$TMP1 ;REPORT A VALID TEST FAILURE.
2648 012234 013737 177740 001640 MOV @#LOADRS,$TMP3
2649 012242 013737 177742 001642 MOV @#HIADRS,$TMP4
2650 012250 005726 TST (SP)+
2651 012252 022626 CMP (SP)+,(SP)+
2652 012254 104033 ERROR 33
2653 012256 012737 177777 177744 MOV #-1,@#MEMERR
2654
2655 012264 006337 012336 SS10: ASL SSMASK ;ROTATE MASK TO FLOAT 0
2656 012270 006137 012340 ROL SSMASK+2 ;TO THE LEFT.
2657 012274 005305 DEC R5
2658 012276 001402 BEQ 1$
2659 012300 000137 011704 JMP SS4
2660 012304 005037 177572 1$: CLR @#MMR0 ;TURN OF MEMORY MANAGEMENT
2661 012310 005037 172516 CLR @#MMR3 ;AND THE MAPPING BOX.
2662 012314 020427 000025 CMP R4,#25 ;IS THE TEST DONE?
2663 012320 002012 BGE SS11 ;YES
2664 012322 000137 011642 JMP SS2 ;NO
2665
2666 012326 000000 SSADR1: .WORD 0 ;USED TO GENERATE THE
2667 012330 000000 .WORD 0 ;TEST ADDRESSES.
2668 012332 000000 SSADR2: .WORD 0
2669
2670 012334 000000 .WORD 0
2671 012336 000000 SSMASK: .WORD 0
2672 012340 000000 .WORD 0
2673
2674 012342 126310 SSCNST: .WORD BOTPRG ;CONTAINS THE ADDRESS OF
2675 012344 000000 .WORD 0 ;THE LAST WORD OF THIS PROGRAM.
2676
2677 012346 104414 SS11: RSET ;DONE!
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688 012350 000004 TST11: SCOPE
2689 012352 012737 000004 001702 MOV #4,$TIMES ;;DO 4 ITERATIONS
2690 000011 PP=$TN-1
2691
2692 012360 012737 012606 055572 MOV #TST12,SKAD ;SET THE SKAD REGISTER
2693 ;IN CASE THE TEST ABORTS.
2694 012366 113737 001502 001632 MOVB $TSTNM,$TMP0
2695 012374 012737 055440 000114 MOV #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
2696
2697 012402 012737 000014 177746 PP1: MOV #M1MO,@#CONTRL ;FORCE MISSES TO BOTH GROUPS
2698 012410 104417 SIZE
2699 012412 000000 PPLOAD: .WORD 0 ;LOW ORDER 16-BITS AND
2700 012414 000000 PPHIAD: .WORD 0 ;HIGH ORDER 6-BITS OF THE

```

```

*****
;*TEST 11 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS DUAL ADDRESS TEST
;*
;*THIS TEST PERFORMS A DUAL ADDRESS TEST ON MEMORY LOCATED
;*AT ADDRESSES LESS THAN 160000 (OCT.) OR WITHIN THE FIRST
;*28K. THE PURPOSE IS TO VERIFY THE THE AMX IS WORKING
;*PROPERLY FOR THE LOW ORDER ADDRESS LINES INVOLVED.
*****

```

```

2701                                     ;HIGHEST WORD ADDRESS IN
2702                                     ;MEMORY.
2703 012416 012737 157776 012602      MOV    #157776,PPLIM      ;ESTABLISH THE UPPER LIMIT
2704 012424 005737 012414              TST    PPHIAD          ;FOR THE TEST.
2705 012430 001007                      BNE    PP2
2706 012432 023737 012602 012412      CMP    PPLIM,PPLOAD
2707 012440 003403                      BLE    PP2
2708 012442 013737 012412 012602      MOV    PPLOAD,PPLIM
2709
2710 012450 012700 126310              PP2:  MOV    #BOTPRG,R0      ;THE LOW LIMIT FOR THIS TEST.
2711 012454 010020                      1$:  MOV    R0,(R0)+        ;WRITE THE ADDRESS IN THE
2712 012456 020037 012602              CMP    R0,PPLIM        ;ADDRESS.
2713 012462 101774                      BLOS  1$
2714
2715 012464 012700 126310              PP3:  MOV    #BOTPRG,R0
2716 012470 011001                      MOV    (R0),R1          ;GO BACK AND READ BACK THE
2717 012472 020001                      CMP    R0,R1           ;ADDRESS, CHECK IT AND
2718 012474 001411                      BEQ    PP4             ;WRITE BACK THE COMPLIMENT.
2719 012476 010037 001644              MOV    R0,$TMP5
2720                                     ;REPORT ERROR.
2721 012502 010137 001636              MOV    R1,$TMP2
2722 012506 010037 001640              MOV    R0,$TMP3
2723 012512 005037 001642              CLR    $TMP4
2724 012516 104034                      1$:  ERROR 34
2725
2726 012520 005120                      PP4:  COM    (R0)+        ;WRITE BACK COMPLIMENT.
2727 012522 020037 012602              CMP    R0,PPLIM
2728 012526 101760                      BLOS  PP3
2729
2730 012530 012700 126310              PP5:  MOV    #BOTPRG,R0      ;GO BACK AND CHECK
2731 012534 011001                      MOV    (R0),R1        ;THE COMPLIMENTED PATTERNS.
2732 012536 010002                      MOV    R0,R2
2733 012540 005102                      COM    R2
2734 012542 020102                      CMP    R1,R2
2735 012544 001411                      BEQ    PP6
2736 012546 010237 001644              MOV    R2,$TMP5
2737 012552 010137 001636              MOV    R1,$TMP2
2738 012556 010037 001640              MOV    R0,$TMP3
2739 012562 005037 001642              CLR    $TMP4
2740 012566 104034                      1$:  ERROR 34
2741
2742 012570 005120                      PP6:  COM    (R0)+
2743 012572 020037 012602              CMP    R0,PPLIM
2744 012576 001356                      BNE    PP5
2745 012600 000401                      BR     PP7
2746
2747 012602 000000                      PPLIM: .WORD 0
2748
2749 012604 104414                      PP7:  RSET              ;DONE!
2750
2751
2752
2753
2754
2755
2756
    
```

```

:*****
:*TEST 12      CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS DUAL ADDRESS TEST
:*
:*THIS TEST PERFORMS A DUAL ADDRESS TEST IDENTICAL TO
:*TST11, EXCEPT THAT IT IS DONE THROUGH THE MAPPING
    
```

```
2757 ;*BOX HERE THEREBY TESTING THE UNIBUS INPUTS TO THE AMX.
2758 ;*
2759 ;*****
2760 012606 000004 TST12: SCOPE
2761 012610 012737 000002 001702 MOV #2,$TIMES ;:DO 2 ITERATIONS
2762 000012 TT=$TN-1
2763
2764 012616 012737 013246 055572 MOV #TST13,SKAD ;:SET THE SKAD REGISTER
2765 ;:IN CASE THE TEST ABORTS.
2766 012624 113737 001502 001632 MOV $TSTNM,$TMP0
2767 012632 012737 055440 000114 MOV #SPUR,@#CACHVEC ;:EXPECT NO PARITY ERRORS.
2768 012640 104416 MMSKIP
2769 012642 012737 000014 177746 TT1: MOV #M1M0,@#CONTRL ;:FORCE MISSES TO BOTH GROUPS.
2770 012650 104417 SIZE
2771 012652 000000 TTLOAD: .WORD 0 ;:DETERMINE THE HIGHEST
2772 012654 000000 TTHIAD: .WORD 0 ;:WORD IN MEMORY.
2773
2774 012656 012737 157776 013242 MOV #157776,TTLIM ;:DETERMINE THE UPPER LIMIT
2775 012664 005737 012654 TST TTHIAD ;:FOR THE TEST.
2776 012670 001007 BNE TT2
2777 012672 023737 013242 012652 CMP TTLIM,TTLOAD
2778 012700 003403 BLE TT2
2779 012702 013737 012652 013242 MOV TTLOAD,TTLIM
2780 012710 TT2:
2781
2782 012710 012700 172340 MOV #KIPAR0,R0 ;:INITIALLY PUT MEMORY
2783 012714 012701 077406 MOV #77406,R1 ;:MANAGEMENT IN A 'PASSIVE'
2784 012720 012702 172300 MOV #KIPDR0,R2 ;:STATE, THAT IS MAP ALL
2785 012724 012703 000010 MOV #10,R3 ;:VIRTUAL ADDRESSES ON TO
2786 012730 010122 64$: MOV R1,(R2)+ ;:THEMSELVES AS PHYSICAL
2787 012732 077302 SOB R3,64$ ;:ADDRESSES.
2788 012734 005020 CLR (R0)+
2789 012736 012720 000200 MOV #200,(R0)+
2790 012742 012720 000400 MOV #400,(R0)+
2791 012746 012720 000600 MOV #600,(R0)+
2792 012752 012720 001000 MOV #1000,(R0)+
2793 012756 012720 001200 MOV #1200,(R0)+
2794 012762 012720 001400 MOV #1400,(R0)+
2795 012766 012710 177600 MOV #177600,(R0)
2796
2797 012772 012737 000060 172516 MOV #60,@#MMR3 ;:TURN ON MEMORY MANAGEMENT.
2798 013000 012737 000001 77572 MOV #1,@#MMR0
2799 013006 012700 126310 MOV #BOTPRG,R0 ;:INITIALIZE A POINTER.
2800
2801 013012 1$:
2802
2803 013012 010037 170214 MOV R0,@#MAPL03 ;:RELOCATE THE ADDRESS IN
2804 013016 005037 170216 CLR @#MAPH03 ;:R0 TO THE UNIBUS,
2805 013022 012737 170600 172354 MOV #170600,@#KIPAR6 ;:THROUGH THE MAPPING BOX
2806 013030 012701 140000 MOV #140000,R1 ;:TO THE CACHE.
2807
2808
2809 013034 010011 MOV R0,(R1) ;:WRITE THE ADDRESS IN THE
2810 013036 062700 000002 ADD #2,R0 ;:ADDRESS
2811 013042 020037 013242 CMP R0,TTLIM
2812 013046 101761 BLOS 1$
```



```

2813
2814 013050 012700 126310          MOV    #BOTPRG,R0
2815
2816 013054          TT3:
2817
2818 013054 010037 170214          MOV    R0,@#MAPL03      ;RELOCATE THE ADDRESS IN
2819 013060 005037 170216          CLR    @#MAPH03        ;R0 TO THE UNIBUS,
2820 013064 012737 170600 172354    MOV    #170600,@#KIPAR6;THROUGH THE MAPPING BOX
2821 013072 012701 140000          MOV    #140000,R1      ;TO THE CACHE.
2822
2823
2824 013076 011102          MOV    (R1),R2         ;READ BACK THE ADDRESS
2825 013100 020002          CMP    R0,R2           ;AS DATA IN THE LOCATION
2826 013102 001411          BEQ    TT4             ;IT ADDRESSES.
2827 013104 010037 001644          MOV    R0,$TMP5       ;REPORT ERROR IF NOT
2828                                     ;EQUAL.
2829 013110 010237 001636          MOV    R2,$TMP2
2830 013114 010037 001640          MOV    R0,$TMP3
2831 013120 005037 001642          CLR    $TMP4
2832 013124 104035          1$: ERROR 35
2833 013126 005111          TT4: COM (R1)         ;WRITE BACK THE
2834 013130 062700 000002          ADD    #2,R0           ;COMPLIMENTED DATA.
2835 013134 020037 013242          CMP    R0,TTLIM
2836 013140 101745          BLOS  TT3
2837
2838 013142 012700 126310          MOV    #BOTPRG,R0
2839
2840 013146          TT5:
2841
2842 013146 010037 170214          MOV    R0,@#MAPL03      ;RELOCATE THE ADDRESS IN
2843 013152 005037 170216          CLR    @#MAPH03        ;R0 TO THE UNIBUS,
2844 013156 012737 170600 172354    MOV    #170600,@#KIPAR6;THROUGH THE MAPPING BOX
2845 013164 012701 140000          MOV    #140000,R1      ;TO THE CACHE.
2846
2847
2848 013170 011102          MOV    (R1),R2         ;GO BACK AND CHECK
2849 013172 010003          MOV    R0,R3           ;THE COMPLIMENTED PATTERNS.
2850 013174 005103          COM    R3
2851 013176 020203          CMP    R2,R3
2852 013200 001411          BEQ    TT6
2853 013202 010337 001644          MOV    R3,$TMP5       ;REPORT ERROR
2854 013206 010237 001636          MOV    R2,$TMP2
2855 013212 010037 001640          MOV    R0,$TMP3
2856 013216 005037 001642          CLR    $TMP4
2857 013222 104035          1$: ERROR 35
2858
2859 013224 005111          TT6: COM (R1)         ;COMPLIMENT BACK THE DATA.
2860 013226 062700 000002          ADD    #2,R0
2861 013232 020037 013242          CMP    R0,TTLIM
2862 013236 001343          BNE   TT5
2863 013240 000401          BR    TT7
2864
2865 013242 000000          TTLIM: .WORD 0
2866
2867 013244 104414          TT7: RSET             ;DONE!
2868
    
```

2869  
2870  
2871  
2872  
2873  
2874  
2875  
2876  
2877  
2878  
2879  
2880  
2881  
2882  
2883  
2884  
2885  
2886  
2887  
2888  
2889  
2890  
2891  
2892  
2893  
2894  
2895  
2896  
2897  
2898  
2899  
2900  
2901  
2902  
2903  
2904  
2905  
2906  
2907  
2908  
2909  
2910  
2911  
2912  
2913  
2914  
2915  
2916  
2917  
2918  
2919  
2920  
2921  
2922  
2923  
2924

013246	000004		
013250	012737	000040	001702
013256	012737	014420	055572
013264	113737	001502	001632
013272	012737	055440	000114
013300	104416		
013302	012700	172340	
013306	012701	077406	
013312	012702	172300	
013316	012703	000010	
013322	010122		
013324	077302		
013326	005020		
013330	012720	000200	
013334	012720	000400	
013340	012720	000600	
013344	012720	001000	
013350	012720	001200	
013354	012720	001400	
013360	012710	177600	
013364	104417		
013366	000000		
013370	000000		

```

*****
*TEST 13      CACHE ADDRESS MEMORY COMPARATOR TEST
*
*THIS IS A TEST OF THE CACHE ADDRESS MEMORY ADDRESS COMPARATORS.
*THIS IS A CIRCUIT MADE UP OF SIX 74585 CHIPS, THREE FOR EACH
*GROUP. EACH CHIP COMPARES FOUR BITS OF THE ADDRESS ON THE
*ADDRESS MULTIPLEXER, AMX, OUTPUT LINES WITH THE RESPECTIVE
*FOUR BITS FROM THE CACHE ADDRESS MEMORY. TWELVE BITS OF
*THE ADDRESS ARE BROKEN DOWN THUS: BITS 10 THROUGH 13
*FOR THE FIRST COMPARATOR; BITS 14 THROUGH 17 FOR
*THE NEXT; AND BITS 18 THROUGH 21 FOR THE LAST.
*THE METHOD CHOSEN FOR THIS TEST IS TO TAKE EACH
*POSSIBLE 4-BIT INPUT CONDITION FOR A COMPARATOR FROM THE
*ADDRESS MEMORY AND PUT EVERY POSSIBLE 4-BIT COMBINATION
*ON THE AMX SIDE OF THE COMPARATOR. FOR 4-BITS
*THERE ARE 16 (DEC) CONDITIONS. THUS FOR EVERY 4-BIT
*ADDRESS MEMORY INPUT TO THE COMPARATOR THERE ARE
*16 AMX INPUT COMBINATIONS ONE OF WHICH WILL CAUSE
*A MATCH AND MAKE THE REFERENCE A HIT. THE OTHER
*15 SHOULD OF COURSE BE MISSES.
*
*****
TST13: SCOPE
MOV     #40,$TIMES      ;;DO 40 ITERATIONS
                ;;SET THE SKAD REGISTER
                ;;IN CASE THE TEST ABORTS.
MOV     #TST14,SKAD
MOVB   $TSTNM,$TMP0
MOV     #SPUR,@#CACHVEC
MMSKIP
                ;;SEE IF THE SWITCH REGISTER
                ;;REFLECTS THE USERS DESIRE
                ;;TO ELIMINATE EXECUTION OF ANY TESTS
                ;;USING MEMORY MANAGEMENT. IF
                ;;SO GO TO THE NEXT TEST.
MOV     #KIPAR0,R0      ;;INITIALLY PUT MEMORY
MOV     #77406,R1       ;;MANAGEMENT IN A 'PASSIVE'
MOV     #KIPDR0,R2     ;;STATE, THAT IS MAP ALL
MOV     #10,R3          ;;VIRTUAL ADDRESSES ON TO
64$:   MOV     R1,(R2)+  ;;THEMSELVES AS PHYSICAL
        SOB    R3,64$   ;;ADDRESSES.
        CLR   (R0)+
        MOV   #200,(R0)+
        MOV   #400,(R0)+
        MOV   #600,(R0)+
        MOV   #1000,(R0)+
        MOV   #1200,(R0)+
        MOV   #1400,(R0)+
        MOV   #177600,(R0)
ZADLO: .WORD 0          ;;THE HIGHEST ADDRESSABLE
ZADHI: .WORD 0          ;;MEMORY WORD AVAILABLE.
    
```

```

2925
2926 013372 005037 014164 CLR ZFLG1 ;ZFLG1 INDICATES WHICH GROUP
2927 ;IS BEING TESTED.
2928 ;ZFLG1 = 0, TESTING GROUP 0.
2929 ;ZFLG1 = 1, TESTING GROUP 1.
2930 ;TEST GROUP 0 FIRST.
2931
2932 013376 012737 000030 014172 MOV #SOM1,ZGS ;ZGS AND ZGM CONTAIN
2933 013404 012737 000044 014170 MOV #S1M0,ZGM ;PATTERNS TO BE USED IN
2934 ;THE CACHE CONTROL REGISTER.
2935 013412 005037 014166 CLR ZFLG2 ;ZFLG2 INDICATES WHICH
2936 ;4-BIT ADDRESS FIELD, OR
2937 ;WHICH COMPARATOR, IS
2938 ;BEING TESTED.
2939 ;ZFLG2 = 0, BITS 10 THROUGH 13
2940 ;ZFLG2 = 1, BITS 14 THROUGH 17
2941 ;ZFLG2 = 2, BITS 18 THROUGH 21
2942 ;ZFLG2 = 3, DONE!
2943
2944 013416 005737 014166 Z1: TST ZFLG2 ;SEE WHICH COMPARATOR
2945 013422 001010 BNE Z2 ;IS BEING TESTED ON THIS
2946 ;PASS AND PUT THE SIXTEEN
2947 ;POSSIBLE ADDRESSES NEEDED
2948 ;FOR THE TEST IN ZTABLE.
2949 013424 012737 002000 014212 MOV #2000,ZTABLE+4 ;BITS 10-13
2950 013432 005037 014214 CLR ZTABLE+6
2951 013436 004737 014310 JSR PC,ZCMTBL ;CALL ZCMTBL TO FINISH THE TABLE.
2952 013442 000432 BR Z5
2953
2954 013444 022737 000001 014166 Z2: CMP #1,ZFLG2
2955 013452 001010 BNE Z3
2956
2957 013454 012737 040000 014212 MOV #40000,ZTABLE+4 ;BITS 14-17
2958 013462 005037 014214 CLR ZTABLE+6
2959 013466 004737 014310 JSR PC,ZCMTBL ;GET ZCMTBL TO FINISH SETTING
2960 013472 000416 BR Z5 ;UP THE TABLE.
2961
2962 013474 022737 000002 014166 Z3: CMP #2,ZFLG2
2963 013502 001010 BNE Z4
2964
2965 013504 012737 000004 014214 MOV #4,ZTABLE+6 ;BITS 18-21
2966 013512 005037 014212 CLR ZTABLE+4
2967 013516 004737 014310 JSR PC,ZCMTBL
2968 013522 000402 BR Z5
2969
2970 013524 000137 014116 Z4: JMP Z14 ;DONE WITH THIS GROUP.
2971
2972 013530 012701 014176 Z5: MOV #ZTHR,R1
2973 013534 013737 014170 177746 MOV ZGM,@#CONTRL
2974 013542 005711 TST (R1) ;MAKE ZTHR A HIT IN BOTH GROUPS.
2975 013544 013737 014172 177746 MOV ZGS,@#CONTRL
2976 013552 005711 TST (R1)
2977
2978 ;FROM NOW ON SELECT THE GROUP BEING TESTED
2979 ;WHILE MISSING THE OTHER GROUP.
2980 013554 012737 000020 172516 MOV #20,@#MMR3 ;TURN ON MEMORY MANAGEMENT.
    
```

```

2981 013562 012737 000001 177572      MOV      #1,@MMRO      ;22-BIT MODE!
2982
2983 013570 012701 014206      MOV      #ZTABLE,R1    ;INITIALIZE R1 AS A POINTER
2984                                ;TO THE ADDRESS WHICH WILL
2985                                ;BE MADE A HIT.
2986
2987 013574      Z7:
2988                                ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2989
2990
2991
2992 013574 023761 013370 000002      CMP      ZADLO+2,2(R1) ;COMPARE THE HIGH ORDER
2993 013602 001005      BNE     64$            ;PARTS OF ZADLO AND (R1).
2994 013604 023711 013366      CMP      ZADLO,(R1)    ;THEN IF NECESSARY
2995 013610 001002      BNE     64$            ;COMPARE THE LOW ORDER PARTS.
2996
2997 013612 000137 013630      JMP      1$            ;THEY WERE EQUAL!
2998
2999 013616 103402      64$:    BLO     65$
3000 013620 000137 013630      JMP      1$            ;THE FIRST ADDRESS IS LARGER
3001                                ;THAN THE SECOND!
3002 013624 000137 014116      65$:    JMP      Z14     ;THE FIRST IS LESS THAN THE
3003                                ;SECOND.
3004
3005
3006 013630 012702 014206      1$:     MOV      #ZTABLE,R2 ;INITIALIZE A POINTER TO
3007                                ;THE ADDRESSES WHICH WILL
3008                                ;BE FED THROUGH THE COMPARATOR
3009                                ;AGAINST THE ADDRESS POINTED
3010                                ;TO BY THE OTHER POINTER, R1
3011
3012 013634 020102      Z8:     CMP      R1,R2    ;DON'T TEST THE ADDRESS
3013 013636 001511      BEQ     Z12            ;AGAINST ITSELF HERE.
3014
3015 013640      Z9:
3016                                ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3017
3018
3019
3020 013640 023762 013370 000002      CMP      ZADLO+2,2(R2) ;COMPARE THE HIGH ORDER
3021 013646 001005      BNE     64$            ;PARTS OF ZADLO AND (R2).
3022 013650 023712 013366      CMP      ZADLO,(R2)    ;THEN IF NECESSARY
3023 013654 001002      BNE     64$            ;COMPARE THE LOW ORDER PARTS.
3024
3025 013656 000137 013674      JMP      Z10           ;THEY WERE EQUAL!
3026
3027 013662 103402      64$:    BLO     65$
3028 013664 000137 013674      JMP      Z10           ;THE FIRST ADDRESS IS LARGER
3029                                ;THAN THE SECOND!
3030 013670 000137 014074      65$:    JMP      Z13     ;THE FIRST IS LESS THAN THE
3031                                ;SECOND.
3032
3033
3034 013674      Z10:
3035
3036 013674 011103      MOV      (R1),R3      ;GET THE PHYSICAL ADDRESS POINTED
    
```

```

3037 013676 042703 177700      BIC    #177700,R3      ;TO BY R1 AND ESTABLISH
3038 013702 011105              MOV    (R1),R5 ;A VIRTUAL ADDRESS WHICH
3039 013704 016104 000002      MOV    2(R1),R4      ;WILL RELOCATE THROUGH
3040 013710 073427 177772      ASHC  #-6,R4        ;KIPAR6. SETUP KIPAR6 AND
3041 013714 010537 172354      MOV    R5,@#KIPAR6   ;LEAVE THE VIRTUAL ADDRESS
3042 013720 062703 140000      ADD    #140000,R3    ;IN R3.
3043
3044
3045 013724 005713              TST    (R3)
3046 013726 005713              TST    (R3)          ;SEE IF YOU CAN GET A HIT.
3047 013730 032737 000010 177752 BIT    #10,@#HITMIS
3048 013736 001011              BNE    Z11
3049 013740 013737 014164 001634 MOV    ZFLG1,$TMP1   ;NO! REPORT THE FAILURE
3050 013746 011137 001636      MOV    (R1),$TMP2
3051 013752 016137 000002 001640 MOV    2(R1),$TMP3
3052 013760 104026              1$:  ERROR  26
3053
3054 013762              Z11:
3055
3056 013762 011203              MOV    (R2),R3      ;GET THE PHYSICAL ADDRESS POINTED
3057 013764 042703 177700      BIC    #177700,R3   ;TO BY R2 AND ESTABLISH
3058 013770 011205              MOV    (R2),R5 ;A VIRTUAL ADDRESS WHICH
3059 013772 016204 000002      MOV    2(R2),R4    ;WILL RELOCATE THROUGH
3060 013776 073427 177772      ASHC  #-6,R4        ;KIPAR6. SETUP KIPAR6 AND
3061 014002 010537 172354      MOV    R5,@#KIPAR6 ;LEAVE THE VIRTUAL ADDRESS
3062 014006 062703 140000      ADD    #140000,R3  ;IN R3.
3063
3064
3065 014012 000240              NOP
3066 014014 005713              TST    (R3)          ;FOR SCOPING WITH AN OSCILLOSCOPE.
3067 014016 032737 000010 177752 BIT    #10,@#HITMIS ;MAKE SURE THERE IS NO
3068 014024 001416              BEQ    Z12          ;MATCH. A MISS?
3069 014026 013737 014164 001634 MOV    ZFLG1,$TMP1   ;GOT A HIT! SO REPORT
3070 014034 011137 001636      MOV    (R1),$TMP2   ;FAILURE
3071 014040 016137 000002 001640 MOV    2(R1),$TMP3
3072 014046 011237 001642      MOV    (R2),$TMP4
3073 014052 016237 000002 001644 MOV    2(R2),$TMP5
3074 014060 104027              1$:  ERROR  27
3075
3076 014062 062702 000004      Z12:  ADD    #4,R2      ;MOVE POINTER TO NEXT AMX
3077                                ;SIDE COMPARATOR INPUT ADDRESS.
3078 014066 020227 014306      CMP    R2,#ZTABOT   ;DONE?
3079 014072 001260              BNE    Z8           ;BRANCH IF NOT DONE.
3080
3081 014074 062701 000004      Z13:  ADD    #4,R1      ;GO TO THE NEXT ADDRESS
3082 014100 020127 014306      CMP    R1,#ZTABOT   ;IN THE TABLE; OR IS THE
3083 014104 001233              BNE    Z7           ;TEST USING THIS ADDRESS TABLE DONE?
3084                                ;IF NOT GO TO Z7.
3085 014106 005237 014166      INC    ZFLG2        ;IF DONE WITH THESE ADDRESSES
3086 014112 000137 013416      JMP    Z1           ;GO BACK TO COMPUTE THE
3087                                ;NEXT ADDRESS TABLE, THAT IS
3088                                ;CHECK THE NEXT 4-BIT
3089                                ;COMPARATOR
3090 014116 005037 177572      Z14:  CLR    @#MMR0     ;TURN OFF MEMORY MANAGEMENT.
3091 014122 005037 172516      CLR    @#MMR3
3092 014126 005737 014164      TST    ZFLG1        ;SEE IF BOTH GROUPS HAVE
    
```

```

3093 014132 001131          BNE      Z15          ;BEEN TESTED. BRANCH IF YES
3094 014134 005237 014164    INC      ZFLG1        ;OTHERWISE CHANGE THE
3095 014140 012737 000044 014172  MOV     #S1MO,ZGS    ;PATTERNS USED IN THE CACHE
3096 014146 012737 000030 014170  MOV     #SOM1,ZGM    ;CONTROL REGISTER AND GO
3097 014154 005037 014166          CLR     ZFLG2        ;BACK TO TEST GROUP 1.
3098 014160 000137 013416    JMP     Z1
3099
3100 014164 000000          ZFLG1: .WORD 0      ;FLAG WHICH DESIGNATES WHICH
3101                                ;GROUP IS BEING TESTED, 0 OR 1.
3102 014166 000000          ZFLG2: .WORD 0      ;FLAG WHICH DESIGNATES WHICH
3103                                ;COMPARATOR IS BEING TESTED:
3104                                ;0 - BITS 10 THROUGH 13
3105                                ;1 - BITS 14 THROUGH 17
3106                                ;2 - BITS 18 THROUGH 21.
3107
3108 014170 000000          ZGM:    .WORD 0      ;PATTERNS USED IN THE HIT
3109 014172 000000          ZGS:    .WORD 0      ;AND MISS REGISTER.
3110 014174 000000          ;.WORD 0
3111 014176 000000          ZTHR:   .WORD 0
3112 014200 000000          ;.WORD 0
3113
3114 014202 000000          ZTMP1:  .WORD 0      ;TEMPORARY STORAGE LOCATIONS
3115 014204 000000          ZTMP2:  .WORD 0      ;USED BY THE ROUTINE, ZCMTBL,
3116                                ;TO GENERATE THE TEST ADDRESS
3117                                ;TABLE, ZTABLE.
3118
3119 014206 000040          ZTABLE: .BLKW 40     ;THE TEST ADDRESS TABLE.
3120 014306 000000          ZTABOT: .WORD 0      ;PRECISION, 22-BIT, ADDRESSES.
3121
3122                                ;THIS ROUTINE IS CALLED TO GENERATE THE TEST ADDRESS
3123                                ;TABLE, BY A 'JSR PC,ZCMTBL'. IT CLEARS THE FIRST
3124                                ;ENTRY; IT ASSUMES THE THE BASE ADDRESS HAS BEEN
3125                                ;PLACED IN THE SECOND ENTRY BEFORE CONTROL IS PASSED
3126                                ;TO IT; THEN, STARTING WITH THE THIRD ENTRY, IT COMPUTES
3127                                ;EACH ENTRY BY ADDING THE BASE ADDRESS TO THE PRECEEDING
3128                                ;ENTRY.
3129 014310 012701 014206    ZCMTBL: MOV     #ZTABLE,R1 ;ESTABLISH A POINTER TO
3130                                ;THE TABLE.
3131                                CLR     (R1)+          ;CLR THE FIRST ENTRY.
3132 014314 005021          CLR     (R1)+
3133 014316 005021          MOV     #16,R0
3134 014320 012700 000016    1$:    MOV     (R1)+,ZTMP1 ;SAVE THE CURRENT ENTRY
3135 014324 012137 014202    MOV     (R1)+,ZTMP2
3136 014330 012137 014204    ;ADD THE OFFSET TO THE
3137                                ;DOUBLE PRECISION ADDITION, UNSIGNED
3138
3139
3140
3141 014334 013711 014202    MOV     ZTMP1,(R1)
3142 014340 013761 014204 000002  MOV     ZTMP1+2,2(R1)
3143 014346 063711 014212    ADD     ZTABLE+4,(R1)
3144 014352 005561 000002    ADC     2(R1)
3145 014356 063761 014214 000002  ADD     ZTABLE+4+2,2(R1)
3146 014364 077021          SOB     R0,1$        ;LOOP UNTIL ZTABLE IS FILLED.
3147
3148

```

```

3149 014366 012702 000020      MOV      #20,R2
3150 014372 012701 014206      MOV      #ZTABLE,R1
3151 014376 012700 014176      MOV      #ZTHR,R0
3152 014402 042700 176000      BIC      #176000,R0
3153 014406 060021      2$:     ADD      R0,(R1)+
3154 014410 005721      TST      (R1)+
3155 014412 077203      SOB      R2,2$
3156
3157 014414 000207      RTS      PC          ;THE RETURN
3158
3159 014416 104414      Z15:    RSET         ;DONE!
3160
3161
3162      ;*****
3163      ;*TEST 14      CACHE ADDRESS MEMORY COUNT PATTERN TEST
3164      ;*
3165      ;*THIS IS A TEST OF THE ADDRESS MEMORY IN THE CACHE.
3166      ;*EVERY BIT IN THE MEMORY IS TURNED ON AND OFF WITHIN
3167      ;*THE LIMITATIONS OF MEMORY SIZE.  THE MANNER IN WHICH
3168      ;*THIS IS DONE IS TO ATTEMPT TO MAKE EVERY ADDRESS
3169      ;*IN AVAILABLE MEMORY A HIT IN EACH GROUP.
3170      ;*
3171      ;*****
3172 014420 000004      TST14:  SCOPE
3173 014422 012737 000002 001702      MOV      #2,$TIMES      ;;DO 2 ITERATIONS
3174      000014      BB=$TN-1
3175 014430      BBO:
3176      ;SET THE SKAD REGISTER
3177 014430 012737 015450 055572      MOV      #TST15,SKAD    ;IN CASE THE TEST ABORTS.
3178
3179 014436 113737 001502 001632      MOVB     $TSTNM,$TMP0
3180
3181 014444 104416      MMSKIP
3182
3183 014446 104417      SIZE
3184 014450 000000      BBLOAD: .WORD 0
3185 014452 000000      BBHIAD: .WORD 0
3186
3187 014454 005037 015152      CLR      BBFLG1         ;TEST GROUP 0 FIRST.
3188 014460 012737 000034 015162      MOV      #SOMOM1,BBGS
3189 014466 012737 000054 015164      MOV      #S1MOM1,BBGM
3190
3191 014474 012737 055440 000114      BB1:    MOV      #SPUR,@#CACHVEC ;EXPECT NO ERRORS, FOR NOW.
3192 014502 012700 014430      MOV      #BBO,R0        ;MAKE THIS CODE HITS IN
3193 014506 012701 001000      MOV      #1000,R1       ;THE GROUP NOT BEING TESTED.
3194 014512 013737 015162 177746      BB2:    MOV      BBGS,@#CONTRL
3195 014520 005760 002000      TST      2000(R0)
3196 014524 013737 015164 177746      MOV      BBGM,@#CONTRL
3197 014532 005720      TST      (R0)+
3198 014534 077112      SOB      R1,BB2
3199
3200 014536 013700 015162      MOV      BBGS,R0        ;FROM NOW ON FORCE
3201 014542 042700 177717      BIC      #177717,R0     ;SELECT THE GROUP BEING
3202 014546 010037 177746      MOV      R0,@#CONTRL   ;TESTED.
3203
3204 014552 012700 015136      BB3:    MOV      #BBADR1,R0    ;INITIALIZE.
    
```

```

3205 014556 012720 126310      MOV    #BOTPRG,(R0)+    ;CONTAINS THE TEST ADDRESS.
3206 014562 005020              CLR    (R0)+
3207 014564 005020              CLR    (R0)+          ;CONTAINS THE LOGICAL 'OR'
3208 014566 005020              CLR    (R0)+          ;OF FAILING ADDRESSES.
3209 014570 012720 177777      MOV    #-1,(R0)+      ;CONTAINS THE LOGICAL 'AND'
3210 014574 012720 177777      MOV    #-1,(R0)+      ;OF BAD ADDRESSES
3211
3212
3213 014600 012700 172340      MOV    #KIPAR0,R0     ;INITIALLY PUT MEMORY
3214 014604 012701 077406      MOV    #77406,R1     ;MANAGEMENT IN A 'PASSIVE'
3215 014610 012702 172300      MOV    #KIPDR0,R2     ;STATE, THAT IS MAP ALL
3216 014614 012703 000010      MOV    #10,R3        ;VIRTUAL ADDRESSES ON TO
3217 014620 010122              MOV    R1,(R2)+      ;THEMSELVES AS PHYSICAL
3218 014622 077302              SOB    R3,64$        ;ADDRESSES.
3219 014624 005020              CLR    (R0)+
3220 014626 012720 000200      MOV    #200,(R0)+
3221 014632 012720 000400      MOV    #400,(R0)+
3222 014636 012720 000600      MOV    #600,(R0)+
3223 014642 012720 001000      MOV    #1000,(R0)+
3224 014646 012720 001200      MOV    #1200,(R0)+
3225 014652 012720 001400      MOV    #1400,(R0)+
3226 014656 012710 177600      MOV    #177600,(R0)
3227
3228 014662 012737 000020 172516  MOV    #20,@#MMR3     ;TURN ON MEMORY MANAGEMENT.
3229 014670 012737 000001 177572  MOV    #1,@#MMR0
3230
3231 014676 005037 015154      CLR    BBFLG2         ;INITIALIZE THE ERROR
3232 014702 005037 015156      CLR    BBCNT1         ;FLAG AND COUNT.
3233 014706 005037 015160      CLR    BBCNT1+2
3234
3235 014712 012737 015166 000114  MOV    #BBERR1,@#CACHVEC ;PREPARE FOR ERRORS.
3236
3237 014720                      BB4:
3238
3239                      ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3240 014720 023737 014452 015140  CMP    BBLOAD+2,BBADR1+2 ;COMPARE THE HIGH ORDER
3241 014726 001006              BNE    64$           ;PARTS OF BBLOAD AND ARG2.
3242 014730 023737 014450 015136  CMP    BBLOAD,BBADR1   ;COMPARE THE LOW ORDER
3243
3244 014736 001002              BNE    64$           ;PARTS.
3245
3246
3247
3248 014740 000137 014756      JMP    BB5            ;THEY WERE EQUAL!
3249
3250 014744 103402              64$: BLO    65$
3251 014746 000137 015054      JMP    BB7            ;THE FIRST ADDRESS IS LARGER
3252                      ;THAN THE SECOND!
3253 014752 000137 014756      65$: JMP    BB5            ;THE FIRST IS LESS THAN THE
3254                      ;SECOND.
3255
3256
3257 014756 012700 015136      BB5: MOV    #BBADR1,R0   ;SET UP MEMORY MANAGEMENT.
3258
3259 014762 011003              MOV    (R0),R3       ;GET THE PHYSICAL ADDRESS POINTED
3260 014764 042703 177700      BIC    #177700,R3    ;TO BY R0 AND ESTABLISH
    
```



```

3261 014770 011005      MOV      (R0),R5 ;A VIRTUAL ADDRESS WHICH
3262 014772 016004 000002  MOV      2(R0),R4 ;WILL RELOCATE THROUGH
3263 014776 073427 177772  ASHC     #-6,R4 ;KIPAR6. SETUP KIPAR6 AND
3264 015002 010537 172354  MOV      R5,@#KIPAR6 ;LEAVE THE VIRTUAL ADDRESS
3265 015006 062703 140000  ADD      #140000,R3 ;IN R3.
3266
3267
3268 015012 000240      NOP
3269 015014 005713      TST      (R3) ;FOR SCOPING WITH AN OSCILLOSCOPE.
3270 015016 005713      TST      (R3) ;TRY TO GET A HIT.
3271
3272 015020 032737 000010 177752  BIT      #10,@#HITMIS ;WAS IT A HIT?
3273 015026 001004      BNE     BB6 ;BRANCH IF YES, OTHERWISE
3274 ;REPORT ERROR.
3275 015030 013737 015152 001636  MOV      BBFLG1,$TMP2
3276 015036 104036 1$:      ERROR 36
3277
3278 015040 062737 000004 015136  BB6:    ADD      #4,BBADR1 ;MOVE TO NEXT WORD PAIR.
3279 015046 005537 015140      ADC     BBADR1+2
3280 015052 000722      BR     BB4
3281
3282 015054 005737 015154      BB7:    TST      BBFLG2 ;DID AN ERROR OCCUR IN
3283 015060 001410      BEQ     BB8 ;THAT GROUP, IF YES PRINT
3284 015062 112737 000037 001514  MOVVB   #37,$ITEMB ;AN ERROR SUMMARY
3285 015070 013737 015152 001634  MOV     BBFLG1,$TMP1
3286 015076 004737 056354      JSR     PC,ERTYPE
3287
3288 015102 005737 015152      BB8:    TST      BBFLG1 ;HAVE BOTH GROUPS BEEN TESTED?
3289 015106 001157      BNE     BBDONE
3290 015110 012737 000001 015152  MOV     #1,BBFLG1 ;IF NOT, GO BACK AND
3291 015116 012737 000054 015162  MOV     #S1MOM1,BBGS ;TEST GROUP 1
3292 015124 012737 000034 015164  MOV     #SOMOM1,BBGM
3293 015132 000137 014474      JMP     BB1
3294
3295 015136 000000      BBADR1: .WORD 0 ;THE TEST ADDRESS.
3296 015140 000000      .WORD 0
3297 015142 000000      BBADR2: .WORD 0 ;LOGICAL 'OR' OF BAD ADDRESSES.
3298 015144 000000      .WORD 0
3299 015146 000000      BBADR3: .WORD 0 ;LOGICAL 'AND' OF BAD ADDRESSES.
3300 015150 000000      .WORD 0
3301
3302 015152 000000      BBFLG1: .WORD 0 ;FLAG: 1, IF TESTING GROUP 1,
3303 ;OR 0, IF TESTING GROUP 0.
3304 015154 000000      BBFLG2: .WORD 0 ;ERROR FLAG: 0, IF NO ERRORS
3305 ;OCCURRED IN THE TESTED
3306 ;GROUP.
3307 015156 000000      BBCNT1: .WORD 0 ;ERROR COUNT.
3308 015160 000000      .WORD 0
3309
3310 015162 000000      BBGS:   .WORD 0 ;PATTERNS FOR THE CACHE
3311 015164 000000      BBGM:   .WORD 0 ;CONTROL REGISTER
3312
3313 015166      BBERR1:
3314
3315 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3316 015166 023737 177742 015140  CMP     LOADRS+2,BBADR1+2 ;COMPARE THE HIGH ORDER
    
```

```

3317 015174 001006          BNE      64$          ;PARTS OF LOADRS AND ARG2.
3318 015176 023737 177740 015136    CMP      LOADRS, BBADR1 ;COMPARE THE LOW ORDER
3319
3320 015204 001002          BNE      64$          ;PARTS.
3321
3322
3323
3324 015206 000137 015224          JMP      BBERR2      ;THEY WERE EQUAL!
3325
3326 015212 103402          64$:    BLO      65$
3327 015214 000137 055440          JMP      SPUR
3328
3329 015220 000137 055440          65$:    JMP      SPUR
3330
3331
3332
3333 015224 032737 000060 177744    BBERR2: BIT      #60, @MEMERR ;MAKE SURE A CACHE ADDRESS
3334 015232 001002          BNE      BBERR3      ;MEMORY PARITY ERROR OCCURRED.
3335 015234 000137 055440          JMP      SPUR
3336
3337 015240          BBERR3:
3338 015240 013737 015152 001640    MOV      BBFLG1, $TMP3 ;REPORT ERROR.
3339 015246 012637 001636    MOV      (SP)+, $TMP2
3340 015252 005726          TST      (SP)+
3341 015254 013737 177744 001642    MOV      @MEMERR, $TMP4
3342 015262 013737 177740 001650    MOV      @LOADRS, $TMP7
3343 015270 013737 177742 001652    MOV      @HIADRS, $TMP10
3344 015276 013737 015136 001644    MOV      BBADR1, $TMP5
3345 015304 013737 015140 001646    MOV      BBADR1+2, $TMP6
3346 015312 104040          1$:    ERROR      40
3347
3348 015314 053737 015136 015142    BIS      BBADR1, BBADR2 ;COMPUTE LOGICAL 'OR' OF
3349 015322 053737 015140 015144    BIS      BBADR1+2, BBADR2+2 ;BAD ADDRESSES.
3350 015330 005137 015146          COM      BBADR3      ;COMPUT THE LOGICAL 'AND'
3351 015334 043737 015136 015146    BIC      BBADR1, BBADR3 ;OF THE BAD ADDRESSES.
3352 015342 005137 015146          COM      BBADR3
3353 015346 005137 015150          COM      BBADR3+2
3354 015352 043737 015140 015150    BIC      BBADR1+2, BBADR3+2
3355 015360 005137 015150          COM      BBADR3+2
3356
3357 015364 012737 177777 015154    MOV      #-1, BBFLG2 ;SET THE ERROR FLAG.
3358 015372 005237 015156          INC      BBCNT1      ;INCREMENT THE ERROR
3359 015376 005537 015160          ADC      BBCNT1+2    ;COUNT.
3360
3361 015402 012737 015424 000114    MOV      #BBERR4, @CACHVEC ;TRY TO GET THE BAD
3362
3363
3364 015410 013705 177740          MOV      @LOADRS, R5 ;ADDRESS OUT OF THE ADDRESS
3365 015414 042705 176001          BIC      #176001, R5 ;MEMORY.
3366 015420 005715          TST      (R5)
3367 015422 000401          BR       BBERR5
3368 015424 022626          BBERR4: CMP      (SP)+, (SP)+
3369 015426 012737 177777 177744    BBERR5: MOV      #-1, @MEMERR
3370 015434 012737 015166 000114    MOV      #BBERR1, @CACHVEC
3371 015442 000137 015040          JMP      BB6
3372
    
```

3373 015446 104414  
3374  
3375  
3376  
3377  
3378  
3379  
3380  
3381  
3382  
3383  
3384  
3385  
3386  
3387  
3388  
3389  
3390  
3391  
3392  
3393  
3394  
3395  
3396  
3397  
3398  
3399  
3400  
3401  
3402  
3403  
3404  
3405  
3406  
3407  
3408  
3409  
3410  
3411  
3412 015450 000004  
3413 015452 012737 000002 001702  
3414 000015  
3415  
3416 015460 012737 016540 055572  
3417  
3418 015466 113737 001502 001632  
3419 015474 012737 055440 000114  
3420 015502 104416  
3421  
3422 015504 012700 172340  
3423 015510 012701 077406  
3424 015514 012702 172300  
3425 015520 012703 000010  
3426 015524 010122  
3427 015526 077302  
3428 015530 005020

```
BBDONE: RSET ;DONE!  
:*****  
: *TEST 15 CACHE ADDRESS MEMORY PARITY LOGIC TEST  
: *  
: *THIS IS A TEST OF THE PARITY CHECKERS AND PARITY GENERATOR  
: *OF THE CACHE ADDRESS MEMORY. EVERY POSSIBLE ADDRESS TAG,  
: *BITS 21 THROUGH 10, WHICH CAN BE STORED IN THE CACHE  
: *ADDRESS MEMORY IS GENERATED, MADE A HIT AND THE  
: *MAINTENANCE REGISTER IS THEN USED TO FORCE A CACHE ADDRESS  
: *MEMORY PARITY ERROR AT EACH OF THE ADDRESSES  
: *GENERATED. NOTE THAT BITS 9 THROUGH 0 OF THE ADDRESSES  
: *IS NOT OF CONCERN, SO THESE BITS WILL BE THE SAME  
: *FOR EACH ADDRESS; THIS IS BECAUSE ONLY BITS 21 THROUGH  
: *10 ARE STORED IN THE ADDRESS MEMORY THEREFORE ONLY  
: *THESE BITS ARE PARITY CHECKED IN THE CACHE ADDRESS  
: *MEMORY PARITY CHECKERS. ALSO NOTE THAT THE RANGE  
: *OF THE ADDRESSES MUST BE LIMITED TO BETWEEN THE  
: *BOUNDS IMPOSED BY THE HIGHEST AVAILABLE MEMORY WORD  
: *AND THE LAST WORD OF MEMORY USED BY THIS PROGRAM.  
: *THE MANNER IN WHICH THE ERROR WILL BE FORCED  
: *WILL BE TO PUT THE INSTRUCTIONS:  
: * 1$: MOV R4,(R2)  
: * TSTADS: CLR (R2)  
: * RTS PC  
: *AT THE PARTICULAR ADDRESS BEING TESTED, WHERE  
: *'TSTADS' IS THE ADDRESS BEING TESTED. R4 CONTAINS  
: *A PATTERN TO BE LOADED IN THE MAINTENANCE REGISTER  
: *WHICH WILL FORCE AN ERROR IN THE CACHE ADDRESS  
: *MEMORY; R2 CONTAINS THE ADDRESS OF THE MAINTENANCE  
: *REGISTER. NOTE FOR EACH ADDRESS R4 WILL FIRST  
: *BE SUCH AS TO CAUSE AN ERROR IN THE LOW  
: *BYTE ADDRESS PARITY CHECKER THEN AT THE SAME  
: *ADDRESS AN ERROR WILL BE FORCED ON THE HIGH BYTE!  
: *THE SEQUENCE OF TEST ADDRESSES WILL BE GENERATED  
: *TWICE ONCE MAKING THEM HITS IN GROUP 0 THEN  
: *MAKING THEM HITS IN GROUP 1.  
: *  
:*****  
TST15: SCOPE  
MOV #2,$TIMES ;:DO 2 ITERATIONS  
AA=$TN-1  
MOV #TST16,SKAD ;SET THE SKAD REGISTER  
;IN CASE THE TEST ABORTS.  
MOVB $TSTNM,$TMP0  
MOV #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.  
MMSKIP  
MOV #KIPAR0,R0 ;INITIALLY PUT MEMORY  
MOV #77406,R1 ;MANAGEMENT IN A 'PASSIVE'  
MOV #KIPDR0,R2 ;STATE, THAT IS MAP ALL  
MOV #10,R3 ;VIRTUAL ADDRESSES ON TO  
64$: MOV R1,(R2)+ ;THEMSELVES AS PHYSICAL  
SOB R3,64$ ;ADDRESSES.  
CLR (R0)+
```

```

3429 015532 012720 000200      MOV      #200,(R0)+
3430 015536 012720 000400      MOV      #400,(R0)+
3431 015542 012720 000600      MOV      #600,(R0)+
3432 015546 012720 001000      MOV      #1000,(R0)+
3433 015552 012720 001200      MOV      #1200,(R0)+
3434 015556 012720 001400      MOV      #1400,(R0)+
3435 015562 012710 177600      MOV      #177600,(R0)
3436
3437 015566 104417                SIZE
3438 015570 000000                AALOAD: .WORD 0           ;ADDRESS OF THE HIGHEST
3439 015572 000000                AAHIAD: .WORD 0           ;WORD IN MEMORY.
3440 015574 042737 000002 015570      BIC      #2,AALOAD
3441
3442 015602 012700 016400      MOV      #AATMP2,R0       ;ESTABLISH BITS 9 THROUGH
3443 015606 042700 176003      BIC      #176003,R0       ;0 TO BE PART OF ALL
3444 015612 010037 016364      MOV      R0,AAOFST        ;THE TEST ADDRESSES.
3445 015616 005037 016366      CLR      AAOFST+2
3446
3447 015622 012737 000020 172516      MOV      #20,@#MMR3      ;ENABLE 22-BIT MODE
3448 015630 012737 000001 177572      MOV      #1,@#MMR0      ;ADDRESSING
3449
3450 015636 012737 000030 016354      MOV      #SOM1,AAGS      ;TEST GROUP 0 FIRST, AAGS
3451 015644 005037 016350      CLR      AAFLG1          ;CONTAINS A PATTERN TO
3452 015650 012737 001400 016356      MOV      #1400,AAERGS    ;BE PUT IN THE CONTROL
3453 015656 012737 004420 016374      MOV      #4420,AAEXER    ;REGISTER. AAERGS CONTAINS
3454                                     ;A PATTERN FOR THE MAINT. REG.
3455 015664 012737 000001 016362      AA1:    MOV      #1,AAADR1+2 ;AAADR1 CONTAINS BITS
3456 015672 005037 016360      CLR      AAADR1          ;10 THROUGH 22 OF
3457                                     ;THE TEST ADDRESS.
3458                                     ;INITIALIZE IT.
3459 015676 013737 016354 177746      MOV      AAGS,@#CONTRL   ;SELECT THE GROUP BEING
3460                                     ;TESTED. MISS THE OTHER
3461                                     ;GROUP.
3462 015704                AA2:    ;GET THE TEST ADDRESS
3463                                     ;INTO THE AAADR2=AAADR1+AAOFST
3464                ;DOUBLE PRECISION ADDITION, UNSIGNED
3465 015704 013737 016360 016370      MOV      AAADR1,AAADR2
3466 015712 013737 016362 016372      MOV      AAADR1+2,AAADR2+2
3467 015720 063737 016364 016370      ADD      AAOFST,AAADR2
3468 015726 005537 016372                ADC      AAADR2+2
3469 015732 063737 016366 016372      ADD      AAOFST+2,AAADR2+2
3470
3471
3472
3473                                     ;SEE IF THIS ADDRESS
3474                                     ;IS A REAL MEMORY LOCATION
3475                                     ;IF NOT THIS GROUP HAS
3476                                     ;BEEN TESTED.
3477
3478                ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3479 015740 023737 016372 015572      CMP      AAADR2+2,AALOAD+2 ;COMPARE THE HIGH ORDER
3480 015746 001006                BNE      64$             ;PARTS OF AAADR2 AND ARG2.
3481 015750 023737 016370 015570      CMP      AAADR2,AALOAD   ;COMPARE THE LOW ORDER
3482
3483 015756 001002                BNE      64$             ;PARTS.
3484
    
```

```

3485
3486
3487 015760 000137 015776          JMP    AA3          ;THEY WERE EQUAL!
3488
3489 015764 103402          64$:  BLO    65$
3490 015766 000137 016306          JMP    AA8          ;THE FIRST ADDRESS IS LARGER
3491          ;THAN THE SECOND!
3492 015772 000137 015776          65$:  JMP    AA3          ;THE FIRST IS LESS THAN THE
3493          ;SECOND.
3494
3495
3496 015776 012737 000001 016352 AA3:  MOV    #1,AAFLG2      ;THE ADDRESS IS GOOD! SET
3497          ;AAFLG2 TO INDICATE AN
3498          ;ERROR IS BEING FORCED
3499          ;ON THE LOW BYTE.
3500          ;ESTABLISH A VIRTUAL ADDRESS WHICH WILL RELOCATE
3501          ;THROUGH KIPAR6 TO THE TEST ADDRESS.
3502 016004 013703 016370          MOV    AAADR2,R3
3503 016010 013702 016372          MOV    AAADR2+2,R2
3504 016014 162703 000002          SUB    #2,R3
3505 016020 005602          SBC    R2
3506 016022 010300          MOV    R3,R0
3507 016024 042700 177700          BIC    #177700,R0      ;R0 CONTAINS THE VIRTUAL
3508 016030 062700 140000          ADD    #140000,R0      ;ADDRESS.
3509
3510 016034 073227 177772          ASHC   #-6,R2          ;SET KIPAR6
3511 016040 010337 172354          MOV    R3,@#KIPAR6
3512
3513 016044 012737 055440 000114      MOV    #SPUR,@#CACHVEC ;RESET VECTOR CACHVEC IN CASE
3514          ;A PARITY ERROR OCCURS
3515          ;WHILE SETTING UP THE
3516          ;INSTRUCTIONS AT THE TEST
3517          ;ADDRESS.
3518          ;PUT THE INSTRUCTIONS AT
3519          ;THE TEST ADDRESS
3520 016052 012710 010112          MOV    #010112,(R0)    ;010112 = 'MOV R4,(R2)'
3521 016056 012760 005012 000002      MOV    #005012,2(R0)   ;005012 = 'CLR (R2)'
3522 016064 012760 000207 000004      MOV    #000207,4(R0)   ;000207 = 'RTS PC'
3523
3524 016072 005760 000002          TST    2(R0)          ;MAKE THE TEST ADDRESS
3525 016076 005760 000002          TST    2(R0)          ;A HIT IN THE GROUP
3526 016102 032737 000010 177752      BIT    #10,@#HITMIS    ;BEING TESTED!
3527 016110 001016          BNE    AA4
3528
3529 016112 012737 016140 001640      MOV    #1$, $TMP3      ;IF UNABLE TO GET A GIT
3530 016120 013737 016350 001634      MOV    AAFLG1, $TMP1   ;REPORT ERROR!
3531 016126 010037 001636          MOV    R0, $TMP2
3532 016132 062737 000002 001636      ADD    #2, $TMP2
3533 016140 104001          1$:  ERROR 1
3534 016142 000137 016270          JMP    AA7          ;GO TO NEXT TEST ADDRESS.
3535
3536 016146          AA4:
3537          ;THE TEST ADDRESS IS NOW
3538 016146 012737 016404 000114      MOV    #AAERR1,@#CACHVEC ;BEING TESTED. NOW RESET
3539          ;CACHVEC TO GO TO THE EXPECTED
3540          ;ERROR HANDLER
    
```

```

3541 016154 012702 177750      MOV    #MAINT,R2      :SET R2
3542 016160 013704 016356      MOV    AAERGS,R4     :SET R4 WHICH WILL BE
3543 016164 042704 005000      BIC    #5000,R4      :LOADED INTO THE MAINT.
3544                                     :REG SO AS TO FORCE
3545                                     :A LOW BYTE ADDRESS
3546                                     :MEMORY PARITY ERROR
3547                                     :IN THE GROUP BEING
3548                                     :TESTED.
3549 016170 000240      NOP                                     :FOR SCOPING WITH AN OSCILLOSCOPE.
3550 016172 004710      JSR    PC,(R0)       :GO TO THE TEST
3551                                     :ADDRESS!
3552
3553 016174                                     AA5:
3554 016174 013737 016350 001636      MOV    AAFLG1,$TMP2  :RETURN,RTS PC, BACK TO
3555 016202 013737 016370 001640      MOV    AAADR2,$TMP3  :HERE IF THE TEST FAILED
3556 016210 013737 016372 001642      MOV    AAADR2+2,$TMP4 :TO FORCE AN ERROR AT
3557 016216 104136      1$:   ERROR 136      :THE TEST ADDRESS'S LOW
3558                                     :BYTE. REPORT THE FAILURE!
3559 016220                                     AA6:
3560                                     :TRY TO DO THE SAME
3561                                     :THING NOW ONLY FORCE THE
3562                                     :ERROR ON THE ADDRESSES
3563                                     :HIGH BYTE!
3564                                     :THE INSTRUCTIONS ARE
3565 016220 012737 000002 016352      MOV    #2,AAFLG2     :ALREADY AT THE TEST
3566 016226 005760 000002                                     :ADDRESS. BUT MAKE SURE
3567 016232 013704 016356      TST    2(R0)         :IT IS STILL A HIT!
3568 016236 042704 002400      MOV    AAERGS,R4     :SET R4 TO FORCE THE
3569 016242 004710      BIC    #2400,R4      :ERROR ON THE HIGH BYTE.
3570                                     :GO DO THE TEST!
3571 016244                                     AA16:
3572 016244 013737 016350 001636      MOV    AAFLG1,$TMP2  :RETURN,RTS PC, BACK TO HERE
3573 016252 013737 016370 001640      MOV    AAADR2,$TMP3  :IF THE TEST FAILED
3574 016260 013737 016372 001642      MOV    AAADR2+2,$TMP4 :IN TRYING TO FORCE A
3575 016266 104137      1$:   ERROR 137      :ERROR ON THE HIGH BYTE
3576                                     :IN THE ADDRESS MEMORY
3577 016270 062737 002000 016360      ADD    #2000,AAADR1  :INCREMENT BITS 21 THROUGH
3578 016276 005537 016362      ADC    AAADR1+2      :10 OF THE TEST ADDRESS
3579 016302 000137 015704      JMP    AA2           :AND GO TEST THIS NEW
3580                                     :TEST ADDRESS!
3581 016306 005737 016350      AA8:  TST    AAFLG1   :SEE IF BOTH GROUPS HAVE
3582 016312 001111      BNE    AADONE        :BEEN TESTED. IF NOT, GO
3583 016314 012737 004440 016374      MOV    #4440,AAEXER  :BACK TO AA1 TO TEST
3584 016322 012737 000044 016354      MOV    #S1M0,AAGS    :GROUP ONE, OTHERWISE DONE!
3585 016330 012737 000001 016350      MOV    #1,AAFLG1
3586 016336 012737 006000 016356      MOV    #6000,AAERGS
3587 016344 000137 015664      JMP    AA1
3588
3589 016350 000000      AAFLG1: .WORD 0      :A FLAG WHICH INDICATES
3590                                     :WHICH GROUP IS BEING TESTED
3591                                     :1 OR 0
3592 016352 000000      AAFLG2: .WORD 0      :A FLAG WHICH INDICATES
3593                                     :WHETHER THE LOW BYTE (1)
3594                                     :THE HIGH BYTE (2) IS
3595                                     :BEING TESTED.
3596 016354 000000      AAGS:  .WORD 0      :A PATTERN FOR THE CONTROL
    
```

```

3597                                     :REGISTER.
3598 016356 000000 AAERGS: .WORD 0      :PATTERN FOR THE MAINT. REG.
3599 016360 000000 AAADR1: .WORD 0      :BITS 21 THROUGH 10 OF
3600 016362 000000                                     :THE TEST ADDRESS.
3601 016364 000000 AAOFS1: .WORD 0      :BITS 9 THROUGH 0 OF
3602 016366 000000                                     :THE TEST ADDRESS.
3603 016370 000000 AAADR2: .WORD 0      :THE TEST ADDRESS.
3604 016372 000000                                     :'AAADR2 = AAADR1+AAOFS1'
3605 016374 000000 AAEXER: .WORD 0      :EXPECTED ERROR REGISTER
3606 016376 000000 AATMP1: .WORD 0      :THESE ADDRESSES ARE
3607 016400 000000 AATMP2: .WORD 0      :USED TO DETERMINE AAOFS1.
3608 016402 000000                                     :
3609
3610 016404 016666 000002 000004 AAERR1: MOV 2(SP),4(SP) :RESET THE STACK. RECALL THAT THE
3611 016412 012616                                     :TEST ROUTINE WAS JSR'ED TO AND
3612                                     :A PARITY ERROR TRAP BROUGHT CONTROL
3613                                     :BACK!!
3614 016414 023737 016374 177744 CMP AAEXER,@MEMERR :MAKE SURE THE ERROR
3615 016422 001405 BEQ 1$ :WHICH OCCURRED WAS
3616 016424 012737 055440 000114 MOV #SPUR,@CACHVEC :THE EXPECTED ERROR AT
3617 016432 000137 055440 JMP SPUR :THE EXPECTED ADDRESS,
3618                                     :IF NOT GO TO THE
3619                                     :SPURIOUS ERROR HANDLER,
3620                                     :SPUR!
3621 016436 1$:
3622
3623                                     :DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3624 016436 023737 016372 177742 CMP AAADR2+2,LOADRS+2 :COMPARE THE HIGH ORDER
3625 016444 001006 BNE 64$ :PARTS OF AAADR2 AND ARG2.
3626 016446 023737 016370 177740 CMP AAADR2,LOADRS :COMPARE THE LOW ORDER
3627
3628 016454 001002 BNE 64$ :PARTS.
3629
3630
3631
3632 016456 000137 016474 JMP AAERR2 :THEY WERE EQUAL!
3633
3634 016462 103402 64$: BLO 65$
3635 016464 000137 055440 JMP SPUR :THE FIRST ADDRESS IS LARGER
3636                                     :THAN THE SECOND!
3637 016470 000137 055440 65$: JMP SPUR :THE FIRST IS LESS THAN THE
3638                                     :SECOND.
3639
3640
3641 016474 012737 177777 177744 AAERR2: MOV #-1,@MEMERR :IF EVERYTHING WAS
3642                                     :CORRECT, CLR THE ERROR
3643 016502 022626 CMP (SP)+,(SP)+ :REGISTER RESET THE
3644                                     :STACK AND CONTINUE
3645 016504 023727 016352 000002 CMP AAFLG2,#2 :TESTING
3646 016512 001002 BNE 1$
3647 016514 000137 016270 JMP AA7 :TEST THE NEXT ADDRESS
3648 016520 023727 016352 000001 1$: CMP AAFLG2,#1
3649 016526 001002 BNE 2$
3650 016530 000137 016220 JMP AA6 :TEST THE HIGH BYTE OF THIS ADDRESS
3651 016534 000000 2$: HALT :???HOW DID WE GET HERE?
3652
    
```

3653 016536 104414

AADONE: RSET ;DONE!

3654  
3655  
3656  
3657  
3658  
3659  
3660  
3661  
3662  
3663  
3664  
3665  
3666  
3667  
3668  
3669  
3670  
3671  
3672  
3673  
3674  
3675  
3676  
3677  
3678  
3679  
3680  
3681  
3682  
3683  
3684  
3685  
3686  
3687  
3688  
3689  
3690  
3691  
3692  
3693  
3694  
3695  
3696  
3697  
3698  
3699  
3700  
3701  
3702  
3703  
3704  
3705  
3706  
3707  
3708

016540 000004  
016542 012737 000002 001702  
016550 000016  
016550 012737 020162 055572  
016556 012737 055440 000114  
016564 113737 001502 001632  
016572 005037 017650  
016576 104416  
016600 104417  
016602 000000  
016604 000000  
016606 005037 017644  
016612 012737 000034 017666  
016620 012737 000054 017670  
016626 005037 017646  
016632 012700 016550  
016636 012701 001000  
016642 013737 017666 177746  
016650 005760 002000  
016654 013737 017670 177746  
016662 005720  
016664 077112

```

:*****
:*TEST 16      CACHE ADDRESS MEMORY DUAL ADDRESS TEST, UPWARD
:*
:*THIS IS A DUAL ADDRESS TEST OF THE CACHE ADDRESS
:*MEMORY. AS MANY AS POSSIBLE DIFFERENT ADDRESS 'TAGS'
:*ARE STORED IN THE 256 (DEC) ADDRESS LOCATIONS OF THE GROUP
:*BEING TESTED. OBVIOUSLY THE NUMBER OF DIFFERENT ADDRESS
:*TAGS AVAILABLE IS LIMITED BY THE SIZE OF THE MEMORY
:*ON THE SYSTEM. NOTE THAT HERE THE WORD 'TAG' REFERS
:*TO THAT PART OF AN ADDRESS, BITS 10 THROUGH 21,
:*WHICH ARE STORED IN THE CACHE ADDRESS MEMORY. HERE
:*THE ADDRESS MEMORY IS WRITTEN IN THE UPWARD DIRECTION,
:*THAT IS 'TAG' 1 IS WRITTEN FIRST, 'TAG' 2 SECOND ETC.
:*THEN EACH ADDRESS WHICH WAS WRITTEN IS TESTED
:* TO SEE IF IT IS A HIT, THUS MAKING SURE NO
:*'TAG' WAS OVERWRITTEN BY A REFERENCE TO ANOTHER
:*'TAG'. NOTE THAT THIS DOES NOT PERFORM A COMPLETE DUAL
:*ADDRESS TEST ON THE ADDRESS MEMORY, FOR THAT WOULD
:*INVOLVE WRITTING THE 'TAGS' IN THE DOWNWARD DIRECTION
:*AS WELL AS THE UPWARD DIRECTION. THE DOWNWARD
:*WRITING PART OF THIS DUAL ADDRESS TEST IS FOUND
:*IN TST17.
:*
:*****
    
```

```

TST16: SCOPE
      MOV      #2,$TIMES      ;;DO 2 ITERATIONS
UU=$TN-1
UU0:
      MOV      #TST17,SKAD    ;SET THE SKAD REGISTER
                                ;IN CASE THE TEST ABORTS.
      MOV      #SPUR,@#CACHVEC ;AT FIRST EXPECT NO ERRORS
      MOV      $TSTNM,$TMP0
      CLR      UUFLG3         ;ERROR FLAG.
      MMSKIP
      SIZE
UULOAD: .WORD 0              ;ADDRESS OF THE HIGHEST WORD
UUHIAD: .WORD 0              ;IN MEMORY
      CLR      UUFLG1         ;TEST GROUP 0 FIRST.
      MOV      #SOMOM1,UUGS
      MOV      #S1MOM1,UUGM
UU1:  CLR      UUFLG2
      MOV      #UU0,R0
      MOV      #1000,R1      ;CLEAR THE PROGRESS FLAG.
                                ;MAKE THIS CODE HITS, IN
                                ;THE GROUP NOT BEING TESTED.
UU2:  MOV      UUGS,@#CONTRL
      TST     2000(R0)
      MOV      UUGM,@#CONTRL
      TST     (R0)+
      SOB     R1,UU2
    
```



```

3709
3710 016666 013701 017666      MOV    UUGS,R1      ;SELECT THE GROUP BEING TESTED.
3711 016672 042701 177717      BIC    #177717,R1
3712 016676 010137 177746      MOV    R1,@#CONTRL
3713
3714
3715 016702 012700 172340      MOV    #KIPAR0,R0   ;INITIALLY PUT MEMORY
3716 016706 012701 077406      MOV    #77406,R1    ;MANAGEMENT IN A 'PASSIVE'
3717 016712 012702 172300      MOV    #KIPDR0,R2   ;STATE, THAT IS MAP ALL
3718 016716 012703 000010      MOV    #10,R3       ;VIRTUAL ADDRESSES ON TO
3719 016722 010122          64$:  MOV    R1,(R2)+      ;THEMSELVES AS PHYSICAL
3720 016724 077302          SOB    R3,64$       ;ADDRESSES.
3721 016726 005020          CLR    (R0)+
3722 016730 012720 000200      MOV    #200,(R0)+
3723 016734 012720 000400      MOV    #400,(R0)+
3724 016740 012720 000600      MOV    #600,(R0)+
3725 016744 012720 001000      MOV    #1000,(R0)+
3726 016750 012720 001200      MOV    #1200,(R0)+
3727 016754 012720 001400      MOV    #1400,(R0)+
3728 016760 012710 177600      MOV    #177600,(R0)
3729
3730 016764 012737 000020 172516      MOV    #20,@#MMR3   ;TURN ON MEMORY MANAGEMENT.
3731 016772 012737 000001 177572      MOV    #1,@#MMR0
3732
3733 017000 005037 017656          CLR    UUADR2       ;INITIALIZE THE ADDRESSES.
3734 017004 005037 017660          CLR    UUADR2+2
3735 017010 012737 140000 017652      MOV    #140000,UUADR1
3736 017016 005037 017654          CLR    UUADR1+2
3737 017022 012701 000400          MOV    #400,R1      ;COUNTER.
3738 017026 012737 017674 000114      MOV    #UUERR1,@#CACHVEC
3739 017034 012737 000001 017646      MOV    #1,UUFLG2    ;KEEP TRACK OF TEST PROGRESS.
3740 017042
3741          UU3:
3742 017042 013737 017652 017662      ;DOUBLE PRECISION ADDITION, UNSIGNED
3743 017050 013737 017654 017664      MOV    UUADR1,UUADR3
3744 017056 063737 017656 017662      MOV    UUADR1+2,UUADR3+2
3745 017064 005537 017664          ADD    UUADR2,UUADR3
3746 017070 063737 017660 017664      ADC    UUADR3+2
3747          ADD    UUADR2+2,UUADR3+2
3748
3749
3750
3751 017076          UU4:
3752
3753          ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3754 017076 023737 017664 016604      CMP    UUADR3+2,UUADR2+2 ;COMPARE THE HIGH ORDER
3755 017104 001006          BNE    64$          ;PARTS OF UUADR3 AND ARG2.
3756 017106 023737 017662 016602      CMP    UUADR3,UUADR2  ;COMPARE THE LOW ORDER
3757
3758 017114 001002          BNE    64$          ;PARTS.
3759
3760
3761
3762 017116 000137 017150          JMP    UU6          ;THEY WERE EQUAL!
3763
3764 017122 103402          64$:  BLO    65$
    
```



```

3821
3822
3823
3824 017406          UU10:
3825
3826          ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3827 017406 023737 017664 016604  CMP      UUADR3+2,UULOAD+2      ;COMPARE THE HIGH ORDER
3828 017414 001006          BNE      64$                    ;PARTS OF UUADR3 AND ARG2.
3829 017416 023737 017662 016602  CMP      UUADR3,UULOAD        ;COMPARE THE LOW ORDER
3830
3831 017424 001002          BNE      64$                    ;PARTS.
3832
3833
3834
3835 017426 000137 017460          JMP      UU12                    ;THEY WERE EQUAL!
3836
3837 017432 103402          64$:    BLO      65$
3838 017434 000137 017444          JMP      UU11                    ;THE FIRST ADDRESS IS LARGER
3839          ;THAN THE SECOND!
3840 017440 000137 017460          65$:    JMP      UU12                    ;THE FIRST IS LESS THAN THE
3841          ;SECOND.
3842
3843
3844 017444 012737 140000 017652  UU11:  MOV      #140000,UUADR1      ;RESET TO GET A VALID ADDRESS.
3845 017452 005037 017654          CLR      UUADR1+2
3846 017456 000735          BR       UU9
3847
3848 017460 012702 017662          UU12:  MOV      #UUADR3,R2
3849
3850 017464 011203          MOV      (R2),R3                ;GET THE PHYSICAL ADDRESS POINTED
3851 017466 042703 177700          BIC      #177700,R3            ;TO BY R2 AND ESTABLISH
3852 017472 011205          MOV      (R2),R5 ;A VIRTUAL ADDRESS WHICH
3853 017474 016204 000002          MOV      2(R2),R4              ;WILL RELOCATE THROUGH
3854 017500 073427 177772          ASHC     #-6,R4                ;KIPAR6. SETUP KIPAR6 AND
3855 017504 010537 172354          MOV      R5,@#KIPAR6          ;LEAVE THE VIRTUAL ADDRESS
3856 017510 062703 140000          ADD      #140000,R3            ;IN R3.
3857
3858
3859 017514 005713          TST      (R3) ;STILL A HIT?
3860 017516 032737 000010 177752  BIT      #10,@#HITMIS
3861 017524 001012          BNE      UU13
3862          ;NOT A HIT, A DUAL ADDRESSING
3863 017526 013737 017644 001636  MOV      UUFLG1,$TMP2          ;PROBLEM?
3864 017534 013737 017662 001640  MOV      UUADR3,$TMP3
3865 017542 013737 017664 001642  MOV      UUADR3+2,$TMP4
3866 017550 104042          1$:    ERROR 42
3867
3868 017552 062737 002000 017652  UU13:  ADD      #2000,UUADR1
3869 017560 005537 017654          ADC      UUADR1+2
3870 017564 062737 000004 017656  ADD      #4,UUADR2              ;LOOP TO READ NEXT ADDRESS
3871 017572 005301          DEC      R1
3872 017574 001402          BEQ     1$
3873 017576 000137 017352          JMP     UU9
3874 017602 012737 000004 017646  1$:    MOV      #4,UUFLG2
3875 017610 005737 017644          UU14:  TST      UUFLG1 ;TESTED BOTH GROUPS?
3876 017614 001161          BNE     UUDONE ;YES.

```

```

3877 017616 012737 000001 017644      MOV    #1,UUFLG1      ;NO, GO TEST GROUP 1.
3878 017624 012737 000054 017666      MOV    #S1MOM1,UUGS
3879 017632 012737 000034 017670      MOV    #SOMOM1,UUGM
3880 017640 000137 016626      JMP    UU1
3881
3882 017644 000000      UUFLG1: .WORD 0      ;FLAG WHICH DESIGNATES
3883                                     ;WHICH GROUP IS BEING TESTED,
3884                                     ;1 OR 0.
3885 017646 000000      UUFLG2: .WORD 0      ;DESIGNATES HOW FAR THE
3886                                     ;TEST HAS PROGRESSED.
3887 017650 000000      UUFLG3: .WORD 0      ;ERROR DURING TEST UUFLG2=4
3888                                     ;PHASE.
3889 017652 000000      UUADR1: .WORD 0      ;ADDRESS WRITTEN INTO ADDRESS
3890 017654 000000      .WORD 0              ;MEMORY LOCATION
3891 017656 000000      UUADR2: .WORD 0      ;ADDRESS MEMORY LOCATION
3892 017660 000000      .WORD 0              ;BEING TESTED
3893 017662 000000      UUADR3: .WORD 0      ;TEST ADDRESS:UUADR3=UUADR1+UUADR2
3894 017664 000000      .WORD 0
3895
3896 017666 000000      UUGS:   .WORD 0      ;PATTERNS FOR THE CACHE CONTROL
3897 017670 000000      UUGM:   .WORD 0      ;REGISTER.
3898 017672 000000      UUTMP:  .WORD 0
3899
3900 017674 032737 000060 177744  UUERR1: BIT    #60,@MEMERR ;WAS THE ERROR A CACHE ADDRESS
3901 017702 001002      BNE    UUERR2        ;MEMORY PARITY ERROR
3902 017704 000137 055440      JMP    SPUR
3903
3904 017710      UUERR2:              ;REPORT ERROR.
3905 017710 012637 001636      MOV    (SP)+,$TMP2
3906 017714 005726      TST   (SP)+
3907 017716 013737 017644 001640      MOV    UUFLG1,$TMP3
3908 017724 013737 177744 001642      MOV    @MEMERR,$TMP4
3909 017732 013737 017662 001644      MOV    UUADR3,$TMP5
3910 017740 013737 017664 001646      MOV    UUADR3+2,$TMP6
3911 017746 013737 177740 001650      MOV    @LOADRS,$TMP7
3912 017754 013737 177742 001652      MOV    @HIADRS,$TMP10
3913 017762 104043      1$:   ERROR 43
3914
3915 017764 042737 177717 001642      BIC    #177717,$TMP4 ;TRY TO GET THE BAD ADDRESS
3916 017772 013737 177746 017672      MOV    @CONTRL,UUTMP ;OUT OF THE ADDRESS MEMORY.
3917 020000 012737 020030 000114      MOV    #UUERR3,@CACHVEC
3918 020006 013705 177740      MOV    @LOADRS,R5
3919 020012 042705 176001      BIC    #176001,R5
3920 020016 013737 001642 177746      MOV    $TMP4,@CONTRL
3921 020024 005715      TST   (R5)
3922 020026 000401      BR    UUERR4
3923 020030 022626      UUERR3: CMP    (SP)+,(SP)+
3924 020032 012737 177777 177744  UUERR4: MOV    #-1,@MEMERR
3925
3926 020040 013737 017672 177746      MOV    UUTMP,@CONTRL ;RESET THE CONTROL REGISTER.
3927 020046 012737 017674 000114      MOV    #UUERR1,@CACHVEC
3928
3929 020054 023727 017646 000001      CMP    UUFLG2,#1      ;RETURN, USING UUFLG2 TO
3930 020062 001002      BNE    1$            ;DECIDE WHERE.
3931 020064 000137 017244      JMP    UU7
3932 020070 023727 017646 000002  1$:   CMP    UUFLG2,#2
    
```

```

3933 020076 001002      BNE      2$
3934 020100 000137 017316    JMP      UU8
3935 020104 023727 017646 000003 2$:     CMP      UUFLG2,#3
3936 020112 001002      BNE      3$
3937 020114 000137 017552    JMP      UU13
3938 020120 023727 017646 000004 3$:     CMP      UUFLG2,#4
3939 020126 001007      BNE      4$
3940 020130 005737 017650    TST      UUFLG3
3941 020134 001011      BNE      UUDONE
3942 020136 005337 017650    DEC      UUFLG3
3943 020142 000137 017610    JMP      UU14
3944
3945 020146 005737 017646      4$:     TST      UUFLG2
3946 020152 001002      BNE      UUDONE      ;??HALT??
3947 020154 000137 016626    JMP      UU1
3948
3949 020160 104414      UUDONE:RSET      ;DONE!
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976 020162 000004      TST17: SCOPE
3977 020164 012737 000002 001702    MOV      #2,$TIMES      ;:DO 2 ITERATIONS
3978 020172 000017      VV=$TN-1
3979 020172      VV0:
3980
3981 020172 012737 021610 055572    MOV      #TST20,SKAD      ;SET THE SKAD REGISTER
3982                                     ;:IN CASE THE TEST ABORTS.
3983 020200 012737 055440 000114    MOV      #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
3984 020206 113737 001502 001632    MOV      $TSTNM,$TMP0
3985
3986 020214 005037 021276      CLR      VVFLG3      ;CLEAR THE ERROR FLAG.
3987
3988 020220 104416      MMSKIP
    
```

```

:*****
:*TEST 17      CACHE ADDRESS MEMORY DUAL ADDRESS TEST, DOWNWARD
:*
:*THIS IS A DUAL ADDRESS TEST OF THE CACHE ADDRESS
:*MEMORY. AS MANY AS POSSIBLE DIFFERENT ADDRESS 'TAGS'
:*ARE STORED IN THE 256 (DEC) ADDRESS LOCATIONS OF THE GROUP
:*BEING TESTED. OBVIOUSLY THE NUMBER OF DIFFERENT ADDRESS
:*TAGS AVAILABLE IS LIMITED BY THE SIZE OF THE MEMORY
:*ON THE SYSTEM. NOTE THAT HERE THE WORD 'TAG' REFERS
:*TO THAT PART OF AN ADDRESS, BITS 10 THROUGH 21,
:*WHICH ARE STORED IN THE CACHE ADDRESS MEMORY. HERE
:*THE ADDRESS MEMORY IS WRITTEN IN THE DOWNWARD DIRECTION,
:*THAT IS 'TAG' 256 IS WRITTEN FIRST, 'TAG' 255 SECOND ETC.
:*THEN EACH ADDRESS WHICH WAS WRITTEN IS TESTED
:* TO SEE IF IT IS A HIT, THUS MAKING SURE NO
:*'TAG' WAS OVERWRITTEN BY A REFERENCE TO ANOTHER
:*'TAG'. NOTE THAT THIS DOES NOT PERFORM A COMPLETE DUAL
:*ADDRESS TEST ON THE ADDRESS MEMORY, FOR THAT WOULD
:*INVOLVE WRITTING THE 'TAGS' IN THE UPWARD DIRECTION
:*AS WELL AS THE DOWNWARD DIRECTION. THE UPWARD
:*WRITING PART OF THIS DUAL ADDRESS TEST IS FOUND
:*IN TST16.
:*
:*****
    
```

```

3976 020162 000004      TST17: SCOPE
3977 020164 012737 000002 001702    MOV      #2,$TIMES      ;:DO 2 ITERATIONS
3978 020172 000017      VV=$TN-1
3979 020172      VV0:
3980
3981 020172 012737 021610 055572    MOV      #TST20,SKAD      ;SET THE SKAD REGISTER
3982                                     ;:IN CASE THE TEST ABORTS.
3983 020200 012737 055440 000114    MOV      #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
3984 020206 113737 001502 001632    MOV      $TSTNM,$TMP0
3985
3986 020214 005037 021276      CLR      VVFLG3      ;CLEAR THE ERROR FLAG.
3987
3988 020220 104416      MMSKIP
    
```

```

3989
3990 020222 104417
3991 020224 000000
3992 020226 000000
3993
3994 020230 005037 021272
3995 020234 012737 000034 021314
3996 020242 012737 000054 021316
3997
3998 020250 005037 021274
3999 020254 012700 020172
4000 020260 012701 001000
4001
4002 020264 013737 021314 177746
4003 020272 005760 002000
4004 020276 013737 021316 177746
4005 020304 005720
4006 020306 077112
4007
4008 020310 013700 021314
4009 020314 042700 177717
4010 020320 010037 177746
4011
4012
4013 020324 012700 172340
4014 020330 012701 077406
4015 020334 012702 172300
4016 020340 012703 000010
4017 020344 010122
4018 020346 077302
4019 020350 005020
4020 020352 012720 000200
4021 020356 012720 000400
4022 020362 012720 000600
4023 020366 012720 001000
4024 020372 012720 001200
4025 020376 012720 001400
4026 020402 012710 177600
4027
4028 020406 012737 000020 172516
4029 020414 012737 000001 177572
4030
4031 020422 012737 001774 021304
4032 020430 005037 021306
4033 020434 012737 140000 021300
4034 020442 005037 021302
4035 020446 012701 000400
4036 020452 012737 021322 000114
4037 020460 012737 000001 021274
4038
4039 020466
4040
4041 020466 013737 021300 021310
4042 020474 013737 021302 021312
4043 020502 063737 021304 021310
4044 020510 005537 021312
    
```

SIZE  
 VVLOAD: .WORD 0 ;ADDRESS OF THE HIGHEST  
 VVHIAD: .WORD 0 ;WORD IN MEMORY.  
 CLR VVFLG1 ;TEST GROUP 0 FIRST  
 MOV #SOMOM1,VVGS  
 MOV #SIMOM1,VVGM  
 VV1: CLR VVFLG2 ;CLEAR THE PROGRESS FLAG  
 MOV #VV0,R0 ;MAKE THIS CODE HITS IN  
 MOV #1000,R1 ;THE GROUP NOT BEING  
 ;TESTED.  
 VV2: MOV VVGS,@#CONTRL  
 TST 2000(R0)  
 MOV VVGM,@#CONTRL  
 TST (R0)+  
 SOB R1,VV2  
 MOV VVGS,R0 ;FROM NOW ON SELECT  
 BIC #177717,R0 ;THE GROUP BEING TESTED.  
 MOV R0,@#CONTRL  
 MOV #KIPAR0,R0 ;INITIALLY PUT MEMORY  
 MOV #77406,R1 ;MANAGEMENT IN A 'PASSIVE'  
 MOV #KIPDR0,R2 ;STATE, THAT IS MAP ALL  
 MOV #10,R3 ;VIRTUAL ADDRESSES ON TO  
 64\$: MOV R1,(R2)+ ;THEMSELVES AS PHYSICAL  
 SOB R3,64\$ ;ADDRESSES.  
 CLR (R0)+  
 MOV #200,(R0)+  
 MOV #400,(R0)+  
 MOV #600,(R0)+  
 MOV #1000,(R0)+  
 MOV #1200,(R0)+  
 MOV #1400,(R0)+  
 MOV #177600,(R0)  
 MOV #20,@#MMR3 ;TURN ON MEMORY MANAGEMENT.  
 MOV #1,@#MMR0  
 MOV #1774,VVADR2 ;INITIALIZE THE ADDRESSES  
 CLR VVADR2+2  
 MOV #140000,VVADR1  
 CLR VVADR1+2  
 MOV #400,R1 ;A COUNTER.  
 MOV #VVERR1,@#CACHVEC ;EXPECT ERRORS NOW.  
 MOV #1,VVFLG2 ;KEEP TRACK OF TEST PROGRESS.  
 VV3: ;DOUBLE PRECISION ADDITION, UNSIGNED  
 MOV VVADR1,VVADR3  
 MOV VVADR1+2,VVADR3+2  
 ADD VVADR2,VVADR3  
 ADC VVADR3+2

```

4045 020514 063737 021306 021312      ADD      VVADR2+2,VVADR3+2
4046
4047
4048
4049
4050 020522                                VV4:
4051
4052                                ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
4053 020522 023737 021312 020226      CMP      VVADR3+2,VVLOAD+2      ;COMPARE THE HIGH ORDER
4054 020530 001006                                BNE      64$                    ;PARTS OF VVADR3 AND ARG2.
4055 020532 023737 021310 020224      CMP      VVADR3,VVLOAD        ;COMPARE THE LOW ORDER
4056
4057 020540 001002                                BNE      64$                    ;PARTS.
4058
4059
4060
4061 020542 000137 020574                                JMP      VV6                    ;THEY WERE EQUAL!
4062
4063 020546 103402                                64$:   BLO      65$
4064 020550 000137 020560                                JMP      VV5                    ;THE FIRST ADDRESS IS LARGER
4065
4066 020554 000137 020574                                65$:   JMP      VV6                    ;THE FIRST IS LESS THAN THE
4067
4068
4069
4070
4071 020560 012737 140000 021300 VV5:   MOV      #140000,VVADR1      ;RESET TO GET A VALID ADDRESS.
4072 020566 005037 021302                                CLR      VVADR1+2
4073 020572 000735                                BR       VV3
4074
4075 020574 012702 021310                                VV6:   MOV      #VVADR3,R2
4076
4077                                MOV      (R2),R3              ;GET THE PHYSICAL ADDRESS POINTED
4078 020602 042703 177700                                BIC      #177700,R3          ;TO BY R2 AND ESTABLISH
4079 020606 011205                                MOV      (R2),R5              ;A VIRTUAL ADDRESS WHICH
4080 020610 016204 000002                                MOV      2(R2),R4            ;WILL RELOCATE THROUGH
4081 020614 073427 177772                                ASHC     #-6,R4              ;KIPAR6. SETUP KIPAR6 AND
4082 020620 010537 172354                                MOV      R5,@#KIPAR6        ;LEAVE THE VIRTUAL ADDRESS
4083 020624 062703 140000                                ADD      #140000,R3          ;IN R3.
4084
4085
4086 020630 005713                                TST      (R3)                ;GET A HIT AT THE
4087 020632 005713                                TST      (R3)                ;TEST ADDRESS
4088 020634 032737 000010 177752      BIT      #10,@#HITMIS
4089 020642 001012                                BNE      VV7
4090
4091 020644 013737 021272 001636                                MOV      VVFLG1,$TMP2
4092 020652 013737 021310 001640                                MOV      VVADR3,$TMP3
4093 020660 013737 021312 001642                                MOV      VVADR3+2,$TMP4
4094 020666 104041                                1$:   ERROR 41
4095
4096 020670 062737 002000 021300 VV7:   ADD      #2000,VVADR1
4097 020676 005537 021302                                ADC      VVADR1+2
4098 020702 062737 177774 021304      ADD      #-4,VVADR2          ;LOOP TO WRITE NEXT ADDRESS
4099 020710 005301                                DEC      R1
4100 020712 001402                                BEQ     1$
    
```

```

4101 020714 000137 020466          JMP      VV3
4102 020720 012737 000002 021274 1$:  MOV      #2,VVFLG2
4103
4104 020726 013700 021316          MOV      VVGM,R0          ;FROM NOW ON SELECT
4105 020732 042700 177717          BIC      #177717,R0      ;THE GROUP NOT BEING
4106 020736 010037 177746          MOV      R0,@#CONTRL    ;TESTED.
4107
4108 020742 012737 001774 021304 VV8:  MOV      #1774,VVADR2    ;NOW RE-GENERATE ALL THE
4109 020750 005037 021306          CLR      VVADR2+2       ;ADDRESSES MADE HITS IN
4110 020754 012737 140000 021300  MOV      #140000,VVADR1 ;THE ABOVE PORTION OF
4111 020762 005037 021302          CLR      VVADR1+2       ;THE TEST, AND MAKE SURE
4112 020766 012701 000400          MOV      #400,R1        ;THEY ARE STILL HITS.
4113 020772 012737 000003 021274  MOV      #3,VVFLG2
4114 021000
4115          VV9:
;DOUBLE PRECISION ADDITION, UNSIGNED
4116 021000 013737 021300 021310  MOV      VVADR1,VVADR3
4117 021006 013737 021302 021312  MOV      VVADR1+2,VVADR3+2
4118 021014 063737 021304 021310  ADD      VVADR2,VVADR3
4119 021022 005537 021312          ADC      VVADR3+2
4120 021026 063737 021306 021312  ADD      VVADR2+2,VVADR3+2
4121
4122
4123
4124
4125 021034          VV10:
;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
4126
4127
4128 021034 023737 021312 020226  CMP      VVADR3+2,VVLOAD+2 ;COMPARE THE HIGH ORDER
4129 021042 001006          BNE      64$            ;PARTS OF VVADR3 AND ARG2.
4130 021044 023737 021310 020224  CMP      VVADR3,VVLOAD   ;COMPARE THE LOW ORDER
4131
4132 021052 001002          BNE      64$            ;PARTS.
4133
4134
4135
4136 021054 000137 021106          JMP      VV12           ;THEY WERE EQUAL!
4137
4138 021060 103402          64$:  BLO      65$
4139 021062 000137 021072          JMP      VV11           ;THE FIRST ADDRESS IS LARGER
4140
4141 021066 000137 021106          65$:  JMP      VV12           ;THE FIRST IS LESS THAN THE
4142
4143
4144
4145 021072 012737 140000 021300 VV11:  MOV      #140000,VVADR1 ;RESET TO CREATE A VALID
4146 021100 005037 021302          CLR      VVADR1+2       ;ADDRESS
4147 021104 000735          BR       VV9
4148
4149 021106 012702 021310          VV12:  MOV      #VVADR3,R2
4150
4151 021112 011203          MOV      (R2),R3        ;GET THE PHYSICAL ADDRESS POINTED
4152 021114 042703 177700          BIC      #177700,R3     ;TO BY R2 AND ESTABLISH
4153 021120 011205          MOV      (R2),R5 ;A VIRTUAL ADDRESS WHICH
4154 021122 016204 000002          MOV      2(R2),R4       ;WILL RELOCATE THROUGH
4155 021126 073427 177772          ASHC    #-6,R4         ;KIPAR6. SETUP KIPAR6 AND
4156 021132 010537 172354          MOV      R5,@#KIPAR6   ;LEAVE THE VIRTUAL ADDRESS
    
```



```

4157 021136 062703 140000      ADD      #140000,R3      ;IN R3.
4158
4159
4160 021142 005713      TST      (R3)          ;STILL A HIT?
4161 021144 032737 000010 177752      BIT      @#HITMIS
4162 021152 001012      BNE      VV13
4163
4164 021154 013737 021272 001636      MOV      VVFLG1,$TMP2  ;REPORT ERROR.
4165 021162 013737 021310 001640      MOV      VVADR3,$TMP3
4166 021170 013737 021312 001642      MOV      VVADR3+2,$TMP4
4167 021176 104042      1$:      ERROR      42
4168
4169 021200 062737 002000 021300 VV13:    ADD      #2000,VVADR1
4170 021206 005537 021302      ADC      VVADR1+2
4171 021212 062737 177774 021304      ADD      #-4,VVADR2
4172 021220 005301      DEC      R1
4173 021222 001402      BEQ      1$
4174 021224 000137 021000      JMP      VV9
4175 021230 012737 000004 021274 1$:      MOV      #4,VVFLG2
4176 021236 005737 021272      VV14:    TST      VVFLG1        ;TESTED BOTH GROUPS?
4177 021242 001161      BNE      VVDONE        ;YES.
4178 021244 012737 000034 021316      MOV      #SOMOM1,VVGM  ;NO GO TEST GROUP 1.
4179 021252 012737 000054 021314      MOV      #S1MOM1,VVGS
4180 021260 012737 000001 021272      MOV      #1,VVFLG1
4181 021266 000137 020250      JMP      VV1
4182
4183 021272 000000      VVFLG1: .WORD      0      ;0 OR 1, GROUP BEING TESTED.
4184 021274 000000      VVFLG2: .WORD      0      ;TEST PROGRESS FLAG.
4185 021276 000000      VVFLG3: .WORD      0      ;ERROR FLAG.
4186
4187 021300 000000      VVADR1: .WORD      0      ;PATTERN WRITTEN INTO THE ADDRESS
4188 021302 000000      .WORD      0      ;MEMORY LOCATION.
4189 021304 000000      VVADR2: .WORD      0      ;ADDRESS MEMORY LOCATION BEING
4190 021306 000000      .WORD      0      ;TESTED X 4.
4191 021310 000000      VVADR3: .WORD      0      ;TEST ADDRESS.
4192 021312 000000      .WORD      0      ;VVADR3=VVADR2+VVADR1
4193
4194 021314 000000      VVGS:   .WORD      0      ;PATTERNS FOR THE CACHE
4195 021316 000000      VVGM:   .WORD      0      ;CONTROL REGISTER.
4196
4197 021320 000000      VVTMP:  .WORD      0
4198
4199 021322 032737 000060 177744 VVERR1: BIT      #60,@MEMERR ;WAS THE ERROR THAT CAUSED
4200 021330 001002      BNE      VVERR2        ;THE TRAP TO HERE A CACHE
4201 021332 000137 055440      JMP      SPUR          ;ADDRESS MEMORY PARITY ERROR?
4202
4203 021336      VVERR2: ;REPORT ERROR.
4204 021336 012637 001636      MOV      (SP)+,$TMP2
4205 021342 005726      TST      (SP)+
4206 021344 013737 021272 001640      MOV      VVFLG1,$TMP3
4207 021352 013737 177744 001642      MOV      @MEMERR,$TMP4
4208 021360 013737 021310 001644      MOV      VVADR3,$TMP5
4209 021366 013737 021312 001646      MOV      VVADR3+2,$TMP6
4210 021374 013737 177740 001650      MOV      @LOADRS,$TMP7
4211 021402 013737 177742 001652      MOV      @HIADRS,$TMP10
4212 021410 104043      1$:      ERROR      43
    
```

```

4213
4214 021412 042737 177717 001642      BIC    #177717,$TMP4      ;TRY TO GET THE BAD ADDRESS
4215 021420 013737 177746 021320      MOV    @#CONTRL,VVTMP    ;OUT OF THE ADDRESS MEMORY.
4216 021426 012737 021456 000114      MOV    #VVERR3,@#CACHVEC
4217 021434 013705 177740      MOV    @#LOADRS,R5
4218 021440 042705 176001      BIC    #176001,R5
4219 021444 013737 001642 177746      MOV    $TMP4,@#CONTRL
4220 021452 005715      TST    (R5)
4221 021454 000401      BR     VVERR4
4222 021456 022626      VVERR3: CMP    (SP)+,(SP)+
4223 021460 012737 177777 177744      VVERR4: MOV    #-1,@#MEMERR
4224
4225 021466 013737 021320 177746      MOV    VVTMP,@#CONTRL    ;RESET THE CONTRL REGISTER
4226 021474 012737 021322 000114      MOV    #VVERR1,@#CACHVEC
4227 021502 023727 021274 000001      CMP    VVFLG2,#1        ;RETURN, USING VVFLG2 TO
4228 021510 001002      BNE    1$                ;DECIDE WHERE.
4229 021512 000137 020670      JMP    VV7
4230 021516 023727 021274 000002 1$:      CMP    VVFLG2,#2
4231 021524 001002      BNE    2$
4232 021526 000137 020742      JMP    VV8
4233 021532 023727 021274 000003 2$:      CMP    VVFLG2,#3
4234 021540 001002      BNE    3$
4235 021542 000137 021200      JMP    VV13
4236 021546 023727 021274 000004 3$:      CMP    VVFLG2,#4
4237 021554 001007      BNE    4$
4238 021556 005737 021276      TST    VVFLG3
4239 021562 001011      BNE    VVDONE
4240 021564 005337 021276      DEC    VVFLG3
4241 021570 000137 021236      JMP    VV14
4242 021574 005737 021274      4$:    TST    VVFLG2
4243 021600 001002      BNE    VVDONE            ;????HALT???
4244 021602 000137 020250      JMP    VV1
4245
4246 021606 104414      VVDONE: RSET            ;DONE!
4247
4248
4249

```

```

4250 *****
4251 ;*TEST-20      CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ONES TEST
4252 ;*
4253 ;*THIS IS A TEST OF THE BYTE MASK GENERATION LOGIC. THIS
4254 ;*IS A FOUR BIT MASK USED BY MAIN MEMORY WHEN PERFORMING
4255 ;*A WRITE. IT DESIGNATES WHICH BYTES OF THE TWO WORDS OF
4256 ;*DATA ON THE MAIN MEMORY DATA BUS LINES ARE TO
4257 ;*BE WRITTEN. THIS WILL BE A TEST DOING CPU DATOB REFERENCES TO
4258 ;*THE CACHE. THE DATOB WILL WRITE 377 INTO A BACK ROUND PATTERN
4259 ;*OF ZEROES.
4260 *****

```

```

4261 021610 000004      TST20: SCOPE
4262 021612 012737 000010 001702      MOV    #10,$TIMES      ;;DO 10 ITERATIONS
4263      CC=$TN-1
4264      ;SET THE SKAD REGISTER
4265 021620 012737 022374 055572      MOV    #TST21,SKAD    ;IN CASE THE TEST ABORTS.
4266
4267 021626 113737 001502 001632      MOVB   $TSTNM,$TMP0
4268 021634 012737 022100 000114      MOV    #CCERR1,@#CACHVEC

```

```

4269
4270 021642 012737 000014 177746      MOV      #MOM1,@#CONTRL ;FORCE MISSES
4271
4272 021650 012700 022074      MOV      #CCTMP2,R0    ;LOCATE THE TEST SPACE.
4273 021654 042700 000003      BIC      #3,R0
4274 021660 010001      MOV      R0,R1
4275 021662 005010      CC1:    CLR      (R0)        ;TEST MASK 0
4276 021664 005060 000002      CLR      2(R0)
4277 021670 000240      NOP
4278 021672 112711 000377      MOVB     #377,(R1)     ;FOR SCOPING WITH AN OSCILLOSCOPE.
4279 021676 022710 000377      CMP      #377,(R0)
4280 021702 001403      BEQ      CC3
4281 021704 004737 022312      CC2:    JSR      PC,CCERR3
4282 021710 000403      BR       CC4
4283 021712 005760 000002      CC3:    TST      2(R0)
4284 021716 001372      BNE      CC2
4285 021720 062701 000001      CC4:    ADD      #1,R1        ;TEST MASK 1.
4286 021724 005010      CLR      (R0)
4287 021726 005060 000002      CLR      2(R0)
4288 021732 000240      NOP
4289 021734 112711 000377      MOVB     #377,(R1)
4290 021740 022710 177400      CMP      #177400,(R0)
4291 021744 001403      BEQ      CC6
4292 021746 004737 022312      CC5:    JSR      PC,CCERR3
4293 021752 000403      BR       CC7
4294 021754 005760 000002      CC6:    TST      2(R0)
4295 021760 001372      BNE      CC5
4296
4297 021762 062701 000001      CC7:    ADD      #1,R1        ;TEST MASK 2.
4298 021766 005010      CLR      (R0)
4299 021770 005060 000002      CLR      2(R0)
4300 021774 000240      NOP
4301 021776 112711 000377      MOVB     #377,(R1)
4302 022002 022760 000377 000002      CMP      #377,2(R0)
4303 022010 001403      BEQ      CC9
4304 022012 004737 022312      CC8:    JSR      PC,CCERR3
4305 022016 000402      BR       CC10
4306 022020 005710      CC9:    TST      (R0)
4307 022022 001373      BNE      CC8
4308
4309 022024 062701 000001      CC10:   ADD      #1,R1        ;TEST MASK 3.
4310 022030 005010      CLR      (R0)
4311 022032 005060 000002      CLR      2(R0)
4312 022036 000240      NOP
4313 022040 112711 000377      MOVB     #377,(R1)
4314 022044 022760 177400 000002      CMP      #177400,2(R0)
4315 022052 001403      BEQ      CC12
4316 022054 004737 022312      CC11:   JSR      PC,CCERR3
4317 022060 000402      BR       CC13
4318 022062 005710      CC12:   TST      (R0)
4319 022064 001373      BNE      CC11
4320
4321 022066 000137 022372      CC13:   JMP      CCDONE
4322
4323 022072 000000      CCTMP1: .WORD 0
4324 022074 000000      CCTMP2: .WORD 0 ;THE TEST AREA.

```

```

4325 022076 000000 .WORD 0
4326
4327
4328 022100 032737 000002 177744 CCERR1: BIT #2,@MEMERR ;SHOULD BE A MAIN MEMORY
4329 022106 001002 BNE 1$ ;ADDRESS AND CONTROL LINE
4330 022110 000137 055440 JMP SPUR ;PARITY ERROR.
4331 022114 020137 177740 1$: CMP R1,@LOADRS ;ERROR ADDRESS SHOULD BE
4332 022120 001402 BEQ CCERR2 ;TEST ADDRESS.
4333 022122 000137 055440 JMP SPUR
4334 022126 012637 001646 CCERR2: MOV (SP)+,$TMP6
4335 022132 005037 001670 CLR $TMP17
4336 022136 005726 TST (SP)+ ;RESET THE STACK
4337 022140 012737 000044 001672 MOV #44,$TMP20
4338 022146 013737 177740 001640 MOV @LOADRS,$TMP3
4339 022154 013737 177742 001642 MOV @HIADRS,$TMP4
4340 022162 013737 177744 001644 MOV @MEMERR,$TMP5
4341 022170 010037 001646 MOV R0,$TMP6
4342 022174 005037 001650 CLR $TMP7
4343 022200 010037 001662 MOV R0,$TMP14
4344 022204 062737 000002 001662 ADD #2,$TMP14
4345 022212 005037 001664 CLR $TMP15
4346 022216 011037 001652 MOV (R0),$TMP10
4347 022222 016037 000002 001654 MOV 2(R0),$TMP11
4348 022230 010137 001656 MOV R1,$TMP12
4349 022234 005037 001660 CLR $TMP13
4350 022240 104044 64$: ERROR 44
4351 022242 012737 177777 177744 MOV #-1,@MEMERR
4352
4353 022250 010002 MOV R0,R2
4354 022252 020102 CMP R1,R2
4355 022254 001002 BNE 2$
4356 022256 000137 021720 JMP CC4
4357 022262 005202 2$: INC R2
4358 022264 020102 CMP R1,R2
4359 022266 001002 BNE 3$
4360 022270 000137 021762 JMP CC7
4361 022274 005202 3$: INC R2
4362 022276 020102 CMP R1,R2
4363 022300 001002 BNE 4$
4364 022302 000137 022024 JMP CC10
4365 022306 000137 022372 4$: JMP CCDONE
4366
4367
4368 022312 011637 001656 CCERR3: MOV (SP),$TMP12 ;REPORT FAILURE TO WRITE
4369 ;THE CORRECT BYTE
4370 022316 010037 001636 MOV R0,$TMP2
4371 022322 005037 001640 CLR $TMP3
4372 022326 010037 001642 MOV R0,$TMP4
4373 022332 062737 000002 001642 ADD #2,$TMP4
4374 022340 005037 001644 CLR $TMP5
4375 022344 011037 001646 MOV (R0),$TMP6
4376 022350 016037 000002 001650 MOV 2(R0),$TMP7
4377 022356 010137 001652 MOV R1,$TMP10
4378 022362 005037 001654 CLR $TMP11
4379 022366 104046 ERROR 46
4380 022370 000207 RTS PC

```

4381  
4382  
4383  
4384  
4385  
4386  
4387  
4388  
4389  
4390  
4391  
4392  
4393  
4394  
4395  
4396  
4397  
4398  
4399  
4400  
4401  
4402  
4403  
4404  
4405  
4406  
4407  
4408  
4409  
4410  
4411  
4412  
4413  
4414  
4415  
4416  
4417  
4418  
4419  
4420  
4421  
4422  
4423  
4424  
4425  
4426  
4427  
4428  
4429  
4430  
4431  
4432  
4433  
4434  
4435  
4436

022372 104414  
  
  
  
  
  
  
  
  
  
022374 000004  
022376 012737 000010 001702  
000021  
  
022404 012737 023172 055572  
  
022412 113737 001502 001632  
022420 012737 022676 000114  
  
022426 012737 000014 177746  
  
022434 012700 022672  
022440 042700 000003  
022444 010001  
  
022446 012710 177777  
022452 012760 177777 000002  
022460 000240  
022462 105011  
022464 022710 177400  
022470 001403  
022472 004737 023110  
022476 000404  
022500 022760 177777 000002  
022506 001371  
  
022510 005201  
022512 012710 177777  
022516 012760 177777 000002  
022524 000240  
022526 105011  
022530 022710 000377  
022534 001403  
022536 004737 023110  
022542 000404  
022544 022760 177777 000002  
022552 001371  
  
022554 005201  
022556 012710 177777  
022562 012760 177777 000002  
022570 000240  
022572 105011

```
CCDONE: RSET ;DONE!  
  
:*****  
:*TEST 21 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ZEROES TEST  
:*  
:*THIS IS ANOTHER TEST OF THE BYTE MASK GENERATION LIGIC.  
:*HERE CPU DATOB'S WILL MOVE ZEROES INTO A BACKGROUND  
:*PATTERN OF ONES.  
:*  
:*****  
TST21: SCOPE  
MOV #10,$TIMES ;DO 10 ITERATIONS  
FF=$TN-1  
  
MOV #TST22,SKAD ;SET THE SKAD REGISTER  
;IN CASE THE TEST ABORTS.  
  
MOVB $TSTNM,$TMPO  
MOV #FFERR1,@#CACHVEC  
  
MOV #MOM1,@#CONTRL ;FORCE MISSES.  
  
MOV #FFTMP2,R0  
BIC #3,R0  
MOV R0,R1  
  
FF1: MOV #-1,(R0) ;TEST MASK 0  
MOV #-1,2(R0)  
NOP ;FOR SCOPING WITH AN OSCILLOSCOPE.  
CLRB (R1)  
CMP #177400,(R0)  
BEQ FF3  
FF2: JSR PC,FFERR3  
BR FF4  
FF3: CMP #-1,2(R0)  
BNE FF2  
  
FF4: INC R1 ;TEST MASK 1.  
MOV #-1,(R0)  
MOV #-1,2(R0)  
NOP ;FOR SCOPING WITH AN OSCILLOSCOPE.  
CLRB (R1)  
CMP #377,(R0)  
BEQ FF6  
FF5: JSR PC,FFERR3  
BR FF7  
FF6: CMP #-1,2(R0)  
BNE FF5  
  
FF7: INC R1 ;TEST MASK 2.  
MOV #-1,(R0)  
MOV #-1,2(R0)  
NOP ;FOR SCOPING WITH AN OSCILLOSCOPE.  
CLRB (R1)
```

```

4437 022574 022760 177400 000002      CMP      #177400,2(R0)
4438 022602 001403      BEQ      FF9
4439 022604 004737 023110      FF8:    JSR      PC,FFERR3
4440 022610 000403      BR       FF10
4441 022612 022710 177777      FF9:    CMP      #-1,(R0)
4442 022616 001372      BNE      FF8
4443
4444 022620 005201      FF10:   INC      R1          ;TEST MASK 3.
4445 022622 012710 177777      MOV      #-1,(R0)
4446 022626 012760 177777 000002      MOV      #-1,2(R0)
4447 022634 000240      NOP
4448 022636 105011      CLR      (R1)          ;FOR SCOPING WITH AN OSCILLOSCOPE.
4449 022640 022760 000377 000002      CMP      #377,2(R0)
4450 022646 001403      BEQ      FF12
4451 022650 004737 023110      FF11:   JSR      PC,FFERR3
4452 022654 000403      BR       FF13
4453 022656 022710 177777      FF12:   CMP      #-1,(R0)
4454 022662 001372      BNE      FF11
4455
4456 022664 000137 023170      FF13:   JMP      FFDONE
4457
4458 022670 000000      FFTMP1: .WORD    0          ;TEST AREA.
4459 022672 000000      FFTMP2: .WORD    0
4460 022674 000000      .WORD    0
4461
4462
4463 022676 032737 000002 177744      FFERR1: BIT      #2,@#MEMERR ;SHOULD BE A MAIN MEMORY
4464 022704 001002      BNE      1$           ;ADDRESS AND CONTROL LINE
4465 022706 000137 055440      JMP      SPUR         ;PARITY ERROR.
4466 022712 020137 177740      1$:     CMP      R1,@#LOADRS ;ERROR ADDRESS SHOULD BE
4467 022716 001402      BEQ      FFERR2      ;TEST ADDRESS.
4468 022720 000137 055440      JMP      SPUR
4469 022724 012637 001646      FFERR2: MOV      (SP)+,$TMP6
4470 022730 005037 001670      CLR      $TMP17
4471 022734 005726      TST      (SP)+          ;RESET THE STACK
4472 022736 012737 000050 001672      MOV      #50,$TMP20
4473 022744 013737 177740 001640      MOV      @#LOADRS,$TMP3
4474 022752 013737 177742 001642      MOV      @#HIADRS,$TMP4
4475 022760 013737 177744 001644      MOV      @#MEMERR,$TMP5
4476 022766 010037 001646      MOV      R0,$TMP6
4477 022772 005037 001650      CLR      $TMP7
4478 022776 010037 001662      MOV      R0,$TMP14
4479 023002 062737 000002 001662      ADD      #2,$TMP14
4480 023010 005037 001664      CLR      $TMP15
4481 023014 011037 001652      MOV      (R0),$TMP10
4482 023020 016037 000002 001654      MOV      2(R0),$TMP11
4483 023026 010137 001656      MOV      R1,$TMP12
4484 023032 005037 001660      CLR      $TMP13
4485 023036 104050      64$:   ERROR    50
4486 023040 012737 177777 177744      MOV      #-1,@#MEMERR
4487
4488 023046 010002      MOV      R0,R2
4489 023050 020102      CMP      R1,R2
4490 023052 001002      BNE      2$
4491 023054 000137 022510      JMP      FF4
4492 023060 005202      2$:    INC      R2

```



```

4549 023250 012722 077406      64$:  MOV    #77406,(R2)+
4550 023254 010401              MOV    R4,R1
4551 023256 072127 000006      ASH    #6,R1
4552 023262 010125              MOV    R1,(R5)+
4553 023264 005025              CLR    (R5)+
4554 023266 010410              MOV    R4,(R0)
4555 023270 062720 170000      ADD    #170000,(R0)+
4556 023274 062704 000200      ADD    #200,R4
4557 023300 077315              SOB    R3,64$
4558 023302 012710 177600      MOV    #177600,(R0)
4559 023306 012712 077406      MOV    #77406,(R2)
4560
4561 023312 012737 000060 172516      MOV    #60,@#MMR3      ;TURN ON MEMORY MANAGEMENT
4562 023320 012737 000001 177572      MOV    #1,@#MMR0      ;AND THE MAPPING BOX RELOCATION.
4563
4564 023326 012737 000014 177746      MOV    #MOM1,@#CONTRL ;FORCE MISSES TO BOTH GROUPS.
4565
4566 023334 012700 023560      MOV    #EETMP2,R0      ;LOCATE THE TEST SPACE.
4567 023340 042700 000003      BIC    #3,R0
4568 023344 010001              MOV    R0,R1
4569
4570 023346 005010      EE1:  CLR    (R0)              ;TEST MASK 0
4571 023350 005060 000002      CLR    2(R0)
4572 023354 000240              NOP
4573 023356 112711 000377      MOVB   #377,(R1)        ;FOR SCOPING WITH AN OSCILLOSCOPE.
4574 023362 022710 000377      CMP    #377,(R0)
4575 023366 001403      BEQ    EE3
4576 023370 004737 023776      EE2:  JSR    PC,EEERR3
4577 023374 000403      BR     EE4
4578 023376 005760 000002      EE3:  TST    2(R0)
4579 023402 001372      BNE    EE2
4580
4581 023404 062701 000001      EE4:  ADD    #1,R1
4582 023410 005010      CLR    (R0)
4583 023412 005060 000002      CLR    2(R0)
4584 023416 000240              NOP
4585 023420 112711 000377      MOVB   #377,(R1)        ;FOR SCOPING WITH AN OSCILLOSCOPE.
4586 023424 022710 177400      CMP    #177400,(R0)
4587 023430 001403      BEQ    EE6
4588 023432 004737 023776      EE5:  JSR    PC,EEERR3
4589 023436 000403      BR     EE7
4590 023440 005760 000002      EE6:  TST    2(R0)
4591 023444 001372      BNE    EE5
4592
4593 023446 062701 000001      EE7:  ADD    #1,R1
4594 023452 005010      CLR    (R0)
4595 023454 005060 000002      CLR    2(R0)
4596 023460 000240              NOP
4597 023462 112711 000377      MOVB   #377,(R1)        ;FOR SCOPING WITH AN OSCILLOSCOPE.
4598 023466 022760 000377 000002      CMP    #377,2(R0)
4599 023474 001403      BEQ    EE9
4600 023476 004737 023776      EE8:  JSR    PC,EEERR3
4601 023502 000402      BR     EE10
4602 023504 005710      EE9:  TST    (R0)
4603 023506 001373      BNE    EE8
4604

```



```

4605 023510 062701 000001      EE10:  ADD    #1,R1
4606 023514 005010              CLR    (R0)
4607 023516 005060 000002      CLR    2(R0)
4608 023522 000240              NOP
4609 023524 112711 000377      MOVB  #377,(R1)
4610 023530 022760 177400 000002  CMP    #177400,2(R0)
4611 023536 001403              BEQ    EE12
4612 023540 004737 023776      EE11:  JSR    PC,EEERR3
4613 023544 000402              BR     EE13
4614 023546 005710      EE12:  TST    (R0)
4615 023550 001373              BNE    EE11
4616
4617 023552 000137 024056      EE13:  JMP    EEDONE
4618
4619 023556 000000      EETMP1: .WORD  0
4620 023560 000000      EETMP2: .WORD  0
4621 023562 000000      .WORD  0
4622
4623
4624 023564 032737 000002 177744  EEERR1: BIT    #2,@MEMERR      ;SHOULD BE A MAIN MEMORY
4625 023572 001002              BNE    1$                    ;ADDRESS AND CONTROL LINE
4626 023574 000137 055440              JMP    SPUR                  ;PARITY ERROR.
4627 023600 020137 177740      1$:   CMP    R1,@LOADRS        ;ERROR ADDRESS SHOULD BE
4628 023604 001402              BEQ    EEERR2               ;TEST ADDRESS.
4629 023606 000137 055440              JMP    SPUR
4630 023612 012637 001646      EEERR2: MOV    (SP)+,$TMP6
4631 023616 005037 001670              CLR    $TMP17
4632 023622 005726              TST    (SP)+                ;RESET THE STACK
4633 023624 012737 000045 001672  MOV    #45,$TMP20
4634 023632 013737 177740 001640  MOV    @LOADRS,$TMP3
4635 023640 013737 177742 001642  MOV    @HIADRS,$TMP4
4636 023646 013737 177744 001644  MOV    @MEMERR,$TMP5
4637 023654 010037 001646  MOV    R0,$TMP6
4638 023660 005037 001650  CLR    $TMP7
4639 023664 010037 001662  MOV    R0,$TMP14
4640 023670 062737 000002 001662  ADD    #2,$TMP14
4641 023676 005037 001664  CLR    $TMP15
4642 023702 011037 001652  MOV    (R0),$TMP10
4643 023706 016037 000002 001654  MOV    2(R0),$TMP11
4644 023714 010137 001656  MOV    R1,$TMP12
4645 023720 005037 001660  CLR    $TMP13
4646 023724 104045      64$:  ERROR  45
4647 023726 012737 177777 177744  MOV    #-1,@MEMERR
4648
4649 023734 010002              MOV    R0,R2
4650 023736 020102              CMP    R1,R2
4651 023740 001002              BNE    2$
4652 023742 000137 023404              JMP    EE4
4653 023746 005202      2$:   INC    R2
4654 023750 020102              CMP    R1,R2
4655 023752 001002              BNE    3$
4656 023754 000137 023446              JMP    EE7
4657 023760 005202      3$:   INC    R2
4658 023762 020102              CMP    R1,R2
4659 023764 001002              BNE    4$
4660 023766 000137 023510              JMP    EE10
    
```

```

4661 023772 000137 024056      4$:  JMP      EEDONE
4662
4663
4664 023776 011637 001656      EEERR3: MOV    (SP), $TMP12      ;REPORT FAILURE TO WRITE
4665                                     ;THE CORRECT BYTE
4666 024002 010037 001636          MOV    R0, $TMP2
4667 024006 005037 001640          CLR    $TMP3
4668 024012 010037 001642          MOV    R0, $TMP4
4669 024016 062737 000002 001642      ADD    #2, $TMP4
4670 024024 005037 001644          CLR    $TMP5
4671 024030 011037 001646          MOV    (R0), $TMP6
4672 024034 016037 000002 001650      MOV    2(R0), $TMP7
4673 024042 010137 001652          MOV    R1, $TMP10
4674 024046 005037 001654          CLR    $TMP11
4675 024052 104047          ERROR  47
4676 024054 000207          RTS    PC
4677
4678
4679 024056 104414          EEDONE: RSET                    ;DONE!
4680
4681      ;*****
4682      ;*TEST 23      CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATOB ZEROES TEST
4683      ;*
4684      ;*THIS IS ANOTHER TEST OF THE BYTE MASK GENERATION LIGIC.
4685      ;*HERE UNIBUS DATOB'S WILL MOVE ZEROES INTO A BACKGROUND
4686      ;*PATTERN OF ONES.
4687      ;*
4688      ;*****
4689 024060 000004          TST23: SCOPE
4690 024062 012737 000010 001702      MOV    #10, $TIMES      ;;DO 10 ITERATIONS
4691      HH=$TN-1
4692
4693 024070 012737 024760 055572      MOV    #TST24, SKAD     ;SET THE SKAD REGISTER
4694                                     ;IN CASE THE TEST ABORTS.
4695 024076 113737 001502 001632      MOV    $TSTNM, $TMP0
4696
4697 024104 104416          MMSKIP
4698
4699 024106 012737 024464 000114      MOV    #HHERR1, @#CACHVEC
4700
4701
4702 024114 012700 172340          MOV    #KIPAR0, R0      ;SET UP MEMORY MANAGEMENT
4703                                     ;TO RELOCATE EVERYTHING
4704 024120 012702 172300          MOV    #KIPDR0, R2     ;THROUGH THE UNIBUS
4705 024124 012703 000007          MOV    #7, R3          ;MAP PASSIVELY TO MEMORY,
4706 024130 005004          CLR    R4              ;BY PASSIVELY IS MEANT
4707 024132 012705 170200          MOV    #MAPL00, R5     ;THAT ADDRESS ARE
4708                                     ;RELOCATED TO THEMSELVES.
4709 024136 012722 077406      64$:  MOV    #77406, (R2)+
4710 024142 010401          MOV    R4, R1
4711 024144 072127 000006          ASH   #6, R1
4712 024150 010125          MOV    R1, (R5)+
4713 024152 005025          CLR   (R5)+
4714 024154 010410          MOV    R4, (R0)
4715 024156 062720 170000          ADD   #170000, (R0)+
4716 024162 062704 000200          ADD   #200, R4
    
```

4717	024166	077315				SOB	R3,64\$	
4718	024170	012710	177600			MOV	#177600,(R0)	
4719	024174	012712	077406			MOV	#77406,(R2)	
4720								
4721	024200	012737	000060	172516		MOV	#60,@#MMR3	;TURN ON MEMORY MANAGEMENT
4722	024206	012737	000001	177572		MOV	#1,@#MMR0	;AND MAPPING BOX RELOCATION.
4723								
4724	024214	012737	000014	177746		MOV	#MOM1,@#CONTRL	;FORCE MISSES.
4725								
4726	024222	012700	024460			MOV	#HHTMP2,R0	;LOCATE THE TEST SPACE.
4727	024226	042700	000003			BIC	#3,R0	
4728	024232	010001				MOV	R0,R1	
4729								
4730	024234	012710	177777		HH1:	MOV	#-1,(R0)	
4731	024240	012760	177777	000002		MOV	#-1,2(R0)	
4732	024246	000240				NOP		;FOR SCOPING WITH AN OSCILLOSCOPE.
4733	024250	105011				CLRB	(R1)	
4734	024252	022710	177400			CMP	#177400,(R0)	
4735	024256	001403				BEQ	HH3	
4736	024260	004737	024676		HH2:	JSR	PC,HHERR3	
4737	024264	000404				BR	HH4	
4738	024266	022760	177777	000002	HH3:	CMP	#-1,2(R0)	
4739	024274	001371				BNE	HH2	
4740								
4741	024276	005201			HH4:	INC	R1	
4742	024300	012710	177777			MOV	#-1,(R0)	
4743	024304	012760	177777	000002		MOV	#-1,2(R0)	
4744	024312	000240				NOP		;FOR SCOPING WITH AN OSCILLOSCOPE.
4745	024314	105011				CLRB	(R1)	
4746	024316	022710	000377			CMP	#377,(R0)	
4747	024322	001403				BEQ	HH6	
4748	024324	004737	024676		HH5:	JSR	PC,HHERR3	
4749	024330	000404				BR	HH7	
4750	024332	022760	177777	000002	HH6:	CMP	#-1,2(R0)	
4751	024340	001371				BNE	HH5	
4752								
4753	024342	005201			HH7:	INC	R1	
4754	024344	012710	177777			MOV	#-1,(R0)	
4755	024350	012760	177777	000002		MOV	#-1,2(R0)	
4756	024356	000240				NOP		;FOR SCOPING WITH AN OSCILLOSCOPE.
4757	024360	105011				CLRB	(R1)	
4758	024362	122760	177400	000002		CMPB	#177400,2(R0)	
4759	024370	001403				BEQ	HH9	
4760	024372	004737	024676		HH8:	JSR	PC,HHERR3	
4761	024376	000403				BR	HH10	
4762	024400	022710	177777		HH9:	CMP	#-1,(R0)	
4763	024404	001372				BNE	HH8	
4764								
4765	024406	005201			HH10:	INC	R1	
4766	024410	012710	177777			MOV	#-1,(R0)	
4767	024414	012760	177777	000002		MOV	#-1,2(R0)	
4768	024422	000240				NOP		;FOR SCOPING WITH AN OSCILLOSCOPE.
4769	024424	105011				CLRB	(R1)	
4770	024426	022760	000377	000002		CMP	#377,2(R0)	
4771	024434	001403				BEQ	HH12	
4772	024436	004737	024676		HH11:	JSR	PC,HHERR3	

```

4773 024442 000403
4774 024444 022710 177777
4775 024450 001372
4776
4777 024452 000137 024756
4778
4779 024456 000000
4780 024460 000000
4781 024462 000000
4782
4783
4784 024464 032737 000002 177744
4785 024472 001002
4786 024474 000137 055440
4787 024500 020137 177740
4788 024504 001402
4789 024506 000137 055440
4790 024512 012637 001646
4791 024516 005037 001670
4792 024522 005726
4793 024524 012737 000051 001672
4794 024532 013737 177740 001640
4795 024540 013737 177742 001642
4796 024546 013737 177744 001644
4797 024554 010037 001646
4798 024560 005037 001650
4799 024564 010037 001662
4800 024570 062737 000002 001662
4801 024576 005037 001664
4802 024602 011037 001652
4803 024606 016037 000002 001654
4804 024614 010137 001656
4805 024620 005037 001660
4806 024624 104051
4807 024626 012737 177777 177744
4808
4809 024634 010002
4810 024636 020102
4811 024640 001002
4812 024642 000137 024276
4813 024646 005202
4814 024650 020102
4815 024652 001002
4816 024654 000137 024342
4817 024660 005202
4818 024662 020102
4819 024664 001002
4820 024666 000137 024406
4821 024672 000137 024756
4822
4823
4824 024676 011637 001656
4825
4826 024702 010037 001636
4827 024706 005037 001640
4828 024712 010037 001642
    
```

```

BR      HH13
HH12:  CMP    #-1,(R0)
      BNE    HH11
      JMP    HH13
      JMP    HH13
      .WORD  0
      .WORD  0
      .WORD  0
      BIT    #2,@#MEMERR
      BNE    1$
      JMP    SPUR
      CMP    R1,@#LOADRS
      BEQ    HHERR2
      JMP    SPUR
      MOV    (SP)+,$TMP6
      CLR    $TMP17
      TST    (SP)+
      MOV    #51,$TMP20
      MOV    @#LOADRS,$TMP3
      MOV    @#HIADRS,$TMP4
      MOV    @#MEMERR,$TMP5
      MOV    R0,$TMP6
      CLR    $TMP7
      MOV    R0,$TMP14
      ADD    #2,$TMP14
      CLR    $TMP15
      MOV    (R0),$TMP10
      MOV    2(R0),$TMP11
      MOV    R1,$TMP12
      CLR    $TMP13
      ERROR  51
      MOV    #-1,@#MEMERR
      MOV    R0,R2
      CMP    R1,R2
      BNE    2$
      JMP    HH4
      INC    R2
      CMP    R1,R2
      BNE    3$
      JMP    HH7
      INC    R2
      CMP    R1,R2
      BNE    4$
      JMP    HH10
      JMP    HH10
      MOV    (SP),$TMP12
      MOV    R0,$TMP2
      CLR    $TMP3
      MOV    R0,$TMP4
    
```

```

;THE TEST AREA
;SHOULD BE A MAIN MEMORY
;ADDRESS AND CONTROL LINE
;PARITY ERROR.
;ERROR ADDRESS SHOULD BE
;TEST ADDRESS.
;RESET THE STACK
;REPORT FAILURE TO WRITE
;THE CORRECT BYTE
    
```

```

4829 024716 062737 000002 001642      ADJ      #2,$TMP4
4830 024724 005037 001644              CLR      $TMP5
4831 024730 011037 001646              MOV      (R0),$TMP6
4832 024734 016037 000002 001650      MOV      2(R0),$TMP7
4833 024742 010137 001652              MOV      R1,$TMP10
4834 024746 005037 001654              CLR      $TMP11
4835 024752 104053              ERROR   53
4836 024754 000207              RTS      PC
4837
4838
4839 024756 104414      HHDONE: RSET              ;DONE!
4840
4841
4842
4843      :*****
4844      :*TEST 24      CACHE ADDRESS MEMORY POWER UP INVALIDATOR TEST
4845      :*
4846      :*THIS TEST IS EXECUTED OPTIONALLY, ON THE CONDITION THAT
4847      :*BIT 12 OF THE SWITCH REGISTER IS ON WHEN PROGRAM CONTROL
4848      :*REACHES THIS POINT. IF THIS SWITCH IS OFF THEN CONTROL
4849      :*IS PASSED TO THE NEXT TEST. THIS IS DONE BECAUSE THIS
4850      :*TEST REQUIRES OPERATOR INTERVENTION. THE USER IS ASKED TO
4851      :*GO THROUGH A POWER DOWN-POWER UP SEQUENCE. THEN
4852      :*A SIMPLE SCAN IS MADE OF MEMORY WHICH CAUSES ALL
4853      :*DATA AND ADDRESS MEMORY LOCATIONS IN THE CACHE TO BE
4854      :*PARITY CHECKED. IF THE POWER UP-CACHE INVLIDATER LOGIC
4855      :*WORKED NO PARITY ERRORS CAN OCCUR. BUT IF THIS INVALIDATER
4856      :*FAILED THERE IS AN EXTREMELY HIGH PROBABILITY FOR THE
4857      :*OCCURENCE OF A CACHE DATA OR CACHE ADDRESS PARITY ERROR.
4858      :*IN FACT IF THE INVALIDATER CIRCUIT IS COMPLETELY INOPERATIVE
4859      :*IT WILL BE VIRTUALLY IMPOSSIBLE TO RESTART THE PROGRAM.
4860      :*WHEREAS MINOR OR NO FAILURES CAN AND WILL BE REPORTED.
4861      :*IF NO PARITY ERRORS ARE ENCOUNTERED THE USER WILL
4862      :*BE NOTIFIED SO THAT HE CAN KNOW IF A FATAL FAILURE
4863      :*HAS OCCURRED.
4864      :*
4865      :*****
4865 024760 000004      TST24: SCOPE
4866      000024      DD=$TN-1
4867
4868 024762 012737 025214 055572      MOV      #TST25,SKAD      ;SET THE SKAD REGISTER
4869      ;IN CASE THE TEST ABORTS.
4870 024770 113737 001502 001632      MOV      $TSTNM,$TMP0
4871 024776 012737 055440 000114      MOV      #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
4872
4873 025004 032777 010000 154526      BIT      #SW12,@SWR      ;SEE IF THE USER HAS CHOSEN
4874 025012 001002              BNE      DD1              ;TO RUN THIS TEST, SW12=1.
4875 025014 000177 030552              JMP      @SKAD            ;NO, SO GO TO NEXT TEST.
4876
4877 025020 012737 025152 000114      DD1:    MOV      #DDPER,@#CACHVEC      ;YES, SO SET UP THE PARITY
4878      ;ERROR VECTOR.
4879 025026 013737 000024 025202      MOV      @#24,DDTMP      ;SAVE THE OLD CONTENTS
4880 025034 012737 025054 000024      MOV      #DDPD,@#24      ;OF THE PWER FAIL TRAP
4881 025042 005037 025204              CLR      DDCNTR          ;VECTOR AND RESET THIS
4882      ;VECTOR. CLEAR AN ERROR COUNT.
4883 025046 104401              TYPE
4884 025050 067140      .WORD   PDMSG1          ;TELL THE USER TO POWER
      ;DOWN.
    
```

```

4885 025052 000777 BR . ;WAIT, SHOULD THIS
4886 ;WAIT TIME OUT????
4887 025054 000240 DDPD: NOP ;FOR SCOPE SYNC!
4888 025056 012737 025066 000024 MOV #DDPV,@#24 ;POWER DOWN ROUTINE
4889 025064 000777 BR . ;JUST SET UP FOR POWER UP.
4890 025066 012706 001400 DDPV: MOV #STACK,SP ;RESET THE STACK POINTER
4891 025072 013737 025202 000024 MOV DDTMP,@#24 ;RESET POWER FAIL VECTOR.
4892 025100 005000 CLR R0 ;SET UP FOR SCAN.
4893 025102 012701 001000 MOV #1000,R1
4894 025106 005720 1$: TST (R0)+
4895 025110 077102 SOB R1,1$
4896 025112 013737 025202 000024 DDPU1: MOV DDTMP,@#24 ;RESET THE POWER FAIL VECTOR.
4897 025120 005737 025204 TST DDCNTR ;WERE THERE ANY ERRORS?
4898 025124 001004 BNE DDPU2
4899 025126 104401 TYPE ;NO
4900 025130 067316 .WORD PDMSG2
4901 025132 000137 025206 JMP DDDONE
4902
4903 025136 DDPU2: ;REPORT ERROR SUMMARY
4904 025136 013737 025204 001636 MOV DDCNTR,$TMP2
4905 025144 104054 1$: ERROR 54
4906 025146 000137 025206 JMP DDDONE
4907
4908 025152 032737 000360 177744 DDPER: BIT #360,@MEMERR ;THE ERROR SHOULD BE
4909 025160 001406 BEQ DDPER1 ;A CACHE ADDRESS OR CACHE
4910 025162 012737 177777 177744 MOV #-1, MEMERR ;DATA PARITY ERROR
4911 025170 005237 025204 INC DDCNTR
4912 025174 000002 RTI
4913
4914 025176 000137 055440 DDPER1: JMP SPUR
4915
4916 025202 000000 DDTMP: .WORD 0 ;STORAGE FOR POWER FAIL
4917 ;VECTORS OLD PC
4918 025204 000000 DDCNTR: .WORD 0 ;ERROR COUNT.
4919
4920 025206 104414 DDDONE: RSET
4921 025210 012706 001400 MOV #STACK,SP
4922
4923 ;*****
4924 ;*TEST 25 CACHE DATA MULTIPLEXER, CDMX, TEST
4925 ;*
4926 ;*THIS TEST PUTS DIFFERENT PATTERNS OF DATA AT THE INPUTS
4927 ;*OF THE CDMX AND TESTS FOR PROPER SELECTION AND GOOD DATA.
4928 ;*
4929 ;*****
4930 025214 000004 TST25: SCOPE
4931 025216 012737 000010 001702 MOV #10,$TIMES ;:DO 10 ITERATIONS
4932 ;SET THE SKAD REGISTER
4933 025224 012737 026322 055572 MOV #TST26,SKAD ;:IN CASE THE TEST ABORTS.
4934
4935 025232 012737 055440 000114 MOV #SPUR,@#CACHVEC ;PREPARE FOR UNEXPECTED ERRORS.
4936 025240 113737 001502 001632 MOV $STNM,$TMP0
4937 025246 012705 000006 MOV #6,R5 ;INITIALIZE
4938 025252 012737 000004 026274 MOV #4,JJCNT
4939 025260 012700 026312 MOV #JJTMP2,R0
4940 025264 042700 176002 BIC #176002,R0
    
```

```

4941 025270 012701 140000      MOV    #TESTR1,R1
4942 025274 060001              ADD    R0,R1
4943 025276 012702 142000      MOV    #TESTR2,R2
4944 025302 060002              ADD    R0,R2
4945 025304 012703 144000      MOV    #TESTR3,R3
4946 025310 060003              ADD    R0,R3
4947 025312 012704 026300      MOV    #JJPAT2,R4
4948
4949 025316 012737 125252 026276      MOV    #125252,JJPAT1 ;JJPAT1 CONTAINS THE DATA
4950                                ;WHICH WILL ENTER THE
4951                                ;MAIN MEMORY EVEN INPUTS
4952                                ;TO THE CDMX. INITIALLY
4953                                ;THIS WILL BE 125252
4954 025324 012737 052525 026300      MOV    #52525,JJPAT2 ;DATA FOR MAIN MEMORY ODD
4955                                ;WORD INPUT TO CDMX
4956 025332 005037 026302              CLR    JJPAT3          ;GROUP 0 DATA INPUTS TO CDMX.
4957 025336 012737 177777 026304      MOV    #-1,JJPAT4     ;GROUP 1 DATA INPUTS TO CDMX.
4958 025344 012737 025344 001510      MOV    #JJ1,$LPERR
4959 025352 013713 026276      MOV    JJPAT1,(R3)    ;WRITE THE MAIN MEMORY
4960 025356 013763 026300 000002      MOV    JJPAT2,2(R3)  ;EVEN AND ODD WORD PATTERNS
4961
4962 025364 012737 000034 177746      MOV    #SOMOM1,@#CONTRL ;WRITE THE GROUP ZERO
4963 025372 013711 026302              MOV    JJPAT3,(R1)   ;PATTERN
4964 025376 013761 026302 177776      MOV    JJPAT3,-2(R1)
4965 025404 013761 026302 000002      MOV    JJPAT3,2(R1)
4966 025412 005711              TST    (R1)
4967 025414 012737 000054 177746      MOV    #S1MOM1,@#CONTRL ;WRITE THE GROUP ONE PATTERN
4968 025422 013712 026304              MOV    JJPAT4,(R2)
4969 025426 013762 026304 177776      MOV    JJPAT4,-2(R2)
4970 025434 013762 026304 000002      MOV    JJPAT4,2(R2)
4971 025442 005712              TST    (R2)
4972
4973 025444 005037 177746              CLR    @#CONTRL
4974 025450 000240              NOP
4975 025452                      JJ2:
4976 025452 000240              NOP
4977 025454 016100 000000      MOV    0(R1),R0
4978 025460 032737 000010 177752      BIT    #10,@#HITMIS ;MUST BE A HIT!
4979 025466 001011              BNE    JJ3
4980 025470 012737 000000 001634      MOV    #0,$TMP1
4981 025476 010137 001636      MOV    R1,$TMP2
4982 025502 062737 000000 001636      ADD    #0,$TMP2
4983 025510 104001 66$:
4984 025512 020037 026302      JJ3:
4985 025516 001406              BEQ    65$
4986 025520 012737 025532 001634      MOV    #64,$TMP1
4987 025526 010037 001636      MOV    R0,$TMP2
4988 025532 104005 64$:
4989 025534 65$:
4990 025534 012737 025542 001510      MOV    #JJ4,$LPERR
4991 025542      JJ4:
4992 025542 000240              NOP
4993 025544 016100 000002      MOV    2(R1),R0
4994 025550 032737 000010 177752      BIT    #10,@#HITMIS ;MUST BE A HIT!
4995 025556 001011              BNE    JJ5
4996 025560 012737 000000 001634      MOV    #0,$TMP1
    
```

4997	025566	010137	001636			MOV	R1,\$TMP2	
4998	025572	062737	000002	001636		ADD	#2,\$TMP2	
4999	025600	104001			66\$:	ERROR	1	
5000	025602	020037	026302		JJ5:	CMP	R0,JJPAT3	
5001	025606	001406				BEQ	65\$	
5002	025610	012737	025622	001634		MOV	#64,\$TMP1	
5003	025616	010037	001636			MOV	R0,\$TMP2	
5004	025622	104005			64\$:	ERROR	5	
5005	025624				65\$:			
5006	025624	012737	025632	001510		MOV	#JJ6,\$LPERR	
5007	025632				JJ6:			
5008	025632	000240				NOP		
5009	025634	016200	000000			MOV	0(R2),R0	
5010	025640	032737	000010	177752		BIT	#10,@#HITMIS	;MUST BE A HIT!
5011	025646	001011				BNE	JJ7	
5012	025650	012737	000001	001634		MOV	#1,\$TMP1	
5013	025656	010237	001636			MOV	R2,\$TMP2	
5014	025662	062737	000000	001636		ADD	#0,\$TMP2	
5015	025670	104001			66\$:	ERROR	1	
5016	025672	020037	026304		JJ7:	CMP	R0,JJPAT4	
5017	025676	001406				BEQ	65\$	
5018	025700	012737	025712	001634		MOV	#64,\$TMP1	
5019	025706	010037	001636			MOV	R0,\$TMP2	
5020	025712	104006			64\$:	ERROR	6	
5021	025714				65\$:			
5022	025714	012737	025722	001510		MOV	#JJ8,\$LPERR	
5023	025722				JJ8:			
5024	025722	000240				NOP		
5025	025724	016200	000002			MOV	2(R2),R0	
5026	025730	032737	000010	177752		BIT	#10,@#HITMIS	;MUST BE A HIT!
5027	025736	001011				BNE	JJ9	
5028	025740	012737	000001	001634		MOV	#1,\$TMP1	
5029	025746	010237	001636			MOV	R2,\$TMP2	
5030	025752	062737	000002	001636		ADD	#2,\$TMP2	
5031	025760	104001			66\$:	ERROR	1	
5032	025762	020037	026304		JJ9:	CMP	R0,JJPAT4	
5033	025766	001406				BEQ	65\$	
5034	025770	012737	026002	001634		MOV	#64,\$TMP1	
5035	025776	010037	001636			MOV	R0,\$TMP2	
5036	026002	104006			64\$:	ERROR	6	
5037	026004				65\$:			
5038	026004	012737	026012	001510		MOV	#JJ10,\$LPERR	
5039	026012	000240			JJ10:	NOP		
5040	026014	012737	000014	177746		MOV	#M1M0,@#CONTRL	;CHECK MAIN MEMORY DATA
5041	026022	011300				MOV	(R3),R0	;EVEN WORD
5042	026024	020037	026276			CMP	R0,JJPAT1	
5043	026030	001403				BEQ	1\$	
5044	026032	010037	001636			MOV	R0,\$TMP2	
5045	026036	104007				ERROR	7	
5046	026040	012737	026046	001510	1\$:	MOV	#JJ11,\$LPERR	
5047	026046	016300	000002		JJ11:	MOV	2(R3),R0	;CHECK MAIN MEMORY EVEN
5048	026052	020037	026300			CMP	R0,JJPAT2	;WORD
5049	026056	001403				BEQ	JJ12	
5050	026060	010037	001636			MOV	R0,\$TMP2	
5051	026064	104010			1\$:	ERROR	10	
5052								



```

5053 026066 005037 177746 JJ12: CLR @#CONTRL
5054 026072 020427 026304 CMP R4,#JJPAT4 ;NOW GET EVERY PERMUTATION
5055 026076 001011 BNE JJ13 ;OF THE FOUR TEST PATTERNS:
5056 ;125252,052525,177777 AND
5057 026100 011437 026306 MOV (R4),JJPAT5 ;000000 INTO JJPAT1, JJPAT2,
5058 026104 013714 026300 MOV JJPAT2,(R4) ;JJPAT3 AND JJPAT4 AND
5059 026110 012704 026300 MOV #JJPAT2,R4 ;REPEAT THE TEST.
5060 026114 013714 026306 MOV JJPAT5,(R4)
5061 026120 000406 BR JJ14
5062
5063 026122 012437 026306 JJ13: MOV (R4)+,JJPAT5
5064 026126 011464 177776 MOV (R4),-2(R4)
5065 026132 013714 026306 MOV JJPAT5,(R4)
5066
5067 026136 005305 JJ14: DEC R5
5068 026140 001402 BEQ 1$
5069 026142 000137 025344 JMP JJ1
5070 026146 012705 000006 1$: MOV #6,R5
5071 026152 013737 026276 026306 MOV JJPAT1,JJPAT5
5072 026160 005337 026274 DEC JJCNT
5073
5074 026164 023727 026274 000003 CMP JJCNT,#3
5075 026172 001010 BNE JJ15
5076 026174 013737 026300 026276 MOV JJPAT2,JJPAT1
5077 026202 013737 026306 026300 MOV JJPAT5,JJPAT2
5078 026210 000137 025344 JMP JJ1
5079
5080 026214 023727 026274 000002 JJ15: CMP JJCNT,#2
5081 026222 001010 BNE JJ16
5082 026224 013737 026302 026276 MOV JJPAT3,JJPAT1
5083 026232 013737 026306 026302 MOV JJPAT5,JJPAT3
5084 026240 000137 025344 JMP JJ1
5085
5086 026244 023727 026274 000001 JJ16: CMP JJCNT,#1
5087 026252 001023 BNE JJ17 ;DONE?
5088 026254 013737 026304 026276 MOV JJPAT4,JJPAT1
5089 026262 013737 026306 026304 MOV JJPAT5,JJPAT4
5090 026270 000137 025344 JMP JJ1
5091
5092 026274 000000 JJCNT: .WORD 0 ;COUNTER USED TO GENERATE
5093 ;PERMUTATIONS.
5094 026276 000000 JJPAT1: .WORD 0 ;MAIN MEMORY EVEN WORD DATA PATTERN
5095 026300 000000 JJPAT2: .WORD 0 ;MAIN MEMORY ODD WORD DATA PATTERN
5096 026302 000000 JJPAT3: .WORD 0 ;GROUP 0 DATA PATTERN
5097 026304 000000 JJPAT4: .WORD 0 ;GROUP 1 DATA PATTERN
5098 026306 000000 JJPAT5: .WORD 0 ;TEMPORARY STORAGE
5099
5100 026310 000000 JJTMP1: .WORD 0 ;TEST AREA, SO CODE WON'T
5101 026312 000000 000000 000000 JJTMP2: .WORD 0,0,0,0 ;OVER LAP THE HITS OF
5102 026320 000000 ;THE TEST WORDS.
5103
5104
5105 026322 JJ17: ;DONE!
5106
5107
5108 ;*****
; *TEST 26 CACHE DATA MEMORY ADDRESS DRIVERS TEST

```

```

5109
5110
5111
5112
5113
5114 026322 000004
5115 026324 012737 000010 001702
5116
5117 026332 012737 027032 055572
5118
5119 026340 012737 055440 000114
5120 026346 113737 001502 001632
5121
5122 026354 012737 000001 027024 GG1:
5123 026362 012737 000054 027026
5124 026370 012737 000034 027030
5125
5126
5127 026376 012700 026376 GG2:
5128 026402 012701 001000
5129 026406 013737 027026 177746 GG3:
5130 026414 005760 002000
5131 026420 013737 027030 177746
5132 026426 005720
5133 026430 077112
5134 026432 013700 027026
5135 026436 042700 177717
5136 026442 010037 177746
5137 026446 012701 140000
5138 026452 012700 001000
5139 026456 012737 026464 001510
5140 026464 000240 GG4:
5141 026466 005011
5142 026470 005711
5143 026472 005711
5144 026474 032737 000010 177752
5145 026502 001006
5146 026504 013737 027024 001634
5147 026512 010137 001636
5148 026516 104001 1$:
5149 026520 005721 2$:
5150 026522 077020
5151 026524 013700 027030
5152 026530 042700 177717
5153 026534 010037 177746
5154
5155 026540 012701 140000
5156 026544 012700 001000
5157 026550 012737 026556 001510
5158 026556 000240 GG5:
5159 026560 010111
5160 026562 005721
5161 026564 077004
5162
5163 026566 012701 140000
5164 026572 012700 001000
    
```

```

: *
: * THIS TEST PERFORMS A DUAL ADDRESS TEST ON THE
: * CACHE DATA MEMORIES OF BOTH GROUPS.
: *
: *****
TST26: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
MOV #TST27,SKAD ;:SET THE SKAD REGISTER
;:IN CASE THE TEST ABORTS.
MOV #SPUR,@#CACHVEC
MOVB $TSTNM,$TMP0
GG1: MOV #1,GGFLG1 ;:INITIALIZE FOR A TEST
MOV #S1MOM1,GGGS ;:ON GROUP 1 FIRST
MOV #SOMOM1,GGGM ;:SOM1 AND S1M0 ARE PATTERNS
;:DESTINED FOR THE CACHE
;:CONTROL REGISTER
GG2: MOV #GG2,R0 ;:MAKE THIS CODE, LOCATIONS
MOV #1000,R1 ;:GG2 THROUGH GG2+2000(OCT),
GG3: MOV GGGS,@#CONTRL ;:HITS IN THE GROUP NOT
TST 2000(R0) ;:BEING TESTED AND MISSES
MOV GGGM,@#CONTRL ;:IN THE GROUP BEING TESTED.
TST (R0)+
SOB R1,GG3
MOV GGGS,R0 ;:MAKE THE TEST AREA
BIC #177717,R0 ;:HITS IN THE GROUP
MOV RO,@#CONTRL ;:BEING TESTED
MOV #TESTR1,R1
MOV #1000,R0
MOV #GG4,$LPERR
GG4: NOP
CLR (R1)
TST (R1)
TST (R1)
BIT #10,@#HITMISS
BNE 2$
MOV GGFLG1,$TMP1
MOV R1,$TMP2
1$: ERROR 1
2$: TST (R1)+
SOB R0,GG4
MOV GGGM,R0 ;:FROM HERE ON SELECT
BIC #177717,R0 ;:THE GROUP NOT BEING
MOV RO,@#CONTRL ;:TESTED
MOV #TESTR1,R1
MOV #1000,R0
MOV #GG5,$LPERR
GG5: NOP
MOV R1,(R1) ;:WRITE #ADDRESS INTO @#ADDRESS.
TST (R1)+
SOB R0,GG5
MOV #TESTR1,R1
MOV #1000,R0
    
```

```

5165 026576 012737 026604 001510      MOV      #GG6,$LPERR
5166 026604 000240      GG6:    NOP
5167 026606 011102      MOV      (R1),R2      ;READ BACK THE ADDRESS
5168 026610 032737 000010 177752      BIT      #10,@#HITMIS
5169 026616 001006      BNE     GG7
5170 026620 013737 027024 001634      MOV      GGFLG1,$TMP1
5171
5172 026626 010137 001636      MOV      R1,$TMP2
5173 026632 104001      1$:    ERROR      1
5174
5175 026634 020102      GG7:    CMP      R1,R2      ;DOES @#ADDRESS CONTAIN
5176 026636 001412      BEQ     GG8          ;#ADDRESS
5177
5178 026640 013737 027024 001634      MOV      GGFLG1,$TMP1
5179 026646 010137 001636      MOV      R1,$TMP2
5180 026652 010237 001640      MOV      R2,$TMP3
5181 026656 010137 001642      MOV      R1,$TMP4
5182 026662 104016      1$:    ERROR      16
5183
5184 026664 005121      GG8:    COM      (R1)+      ;COMPLIMENT DATA
5185 026666 077032      SOB     R0,GG6      ;LOOP FOR NEXT ADDRESS.
5186 026670 012701 140000      MOV      #TSTR1,R1
5187 026674 012700 001000      MOV      #1000,R0
5188 026700 012737 026706 001510      MOV      #GG9,$LPERR
5189 026706 000240      GG9:    NOP
5190 026710 011102      MOV      (R1),R2      ;GO BACK AND CHECK
5191 026712 032737 000010 177752      BIT      #10,@#HITMIS ;COMPLIMENTED DATA
5192 026720 001006      BNE     GG10
5193 026722 013737 027024 001634      MOV      GGFLG1,$TMP1
5194 026730 010137 001636      MOV      R1,$TMP2
5195 026734 104001      1$:    ERROR      1
5196
5197
5198 026736 010103      GG10:   MOV      R1,R3      ;IS COMPLIMENT DATA CORRECT?
5199 026740 005103      COM     R3
5200 026742 020302      CMP     R3,R2
5201 026744 001412      BEQ     GG11
5202 026746 013737 027024 001634      MOV      GGFLG1,$TMP1
5203 026754 010337 001636      MOV      R3,$TMP2
5204 026760 010237 001640      MOV      R2,$TMP3
5205 026764 010137 001642      MOV      R1,$TMP4
5206 026770 104016      1$:    ERROR      16
5207
5208 026772 005721      GG11:   TST     (R1)+      ;TEST NEXT LOCATION
5209 026774 077034      SOB     R0,GG9
5210
5211 026776 012737 000034 027026      MOV      #SOMOM1,GGGS ;GO BACK AND RUN
5212 027004 012737 000054 027030      MOV      #S1MOM1,GGGM ;TEST IN GROUP 0.
5213 027012 005337 027024      DEC     GGFLG1
5214 027016 001005      BNE     GG12
5215 027020 000137 026376      JMP     GG2
5216
5217 027024 000000      GGFLG1: .WORD    0      ;GROUP BEING TESTED, 0 OR 1.
5218
5219 027026 000000      GGGS:   .WORD    0      ;CACHE CONTROL REGISTER
5220 027030 000000      GGGM:   .WORD    0      ;PATTERNS

```

```

5221
5222 027032 GG12: ;DONE!
5223
5224
5225
5226
5227
5228
5229
5230
5231 027032 000004
5232 027034 012737 000010 001702
5233
5234 027042 012737 030012 055572
5235
5236 027050 012737 055440 000114
5237 027056 113737 001502 001632
5238
5239 027064 012737 000001 027520 LL1: MOV #1,LLFLG1 ;TEST GROUP ONE FIRST
5240 027072 012737 000044 027526 MOV #S1M0,LLGS ;S1M0 AND SOM1 ARE PATTERNS
5241 027100 012737 000030 027530 MOV #SOM1,LLGM ;WHICH WILL BE LOADED INTO
5242 027106 012737 027106 001510 LL2: MOV #LL2,$LPERR ;THE CACHE CONTROL REGISTER.
5243 027114 012737 055440 000114 MOV #SPUR,@#CACHVEC
5244 027122 012700 027106 MOV #LL2,R0 ;MAKE THIS CODE, LOCATIONS
5245 027126 012701 001000 MOV #1000,R1 ;LL2 THROUGH LL2+2000 (OCT)
5246 ;HITS IN THE CACHE GROUP
5247 027132 013737 027530 177746 LL3: MOV LLGM,@#CONTRL ;NOT BEING TESTED, AND MISSES
5248 027140 005710 TST (R0) ;TO THE CACHE GROUP BEING
5249 027142 013737 027526 177746 MOV LLGS,@#CONTRL ;TESTED.
5250 027150 005760 002000 TST 2000(R0)
5251 027154 062700 000002 ADD #2,R0
5252 027160 077114 SOB R1,LL3
5253
5254 027162 012701 140000 MOV #TESTR1,R1 ;MAKE THE MEMORY TEST AREA
5255 027166 012700 001000 MOV #1000,R0 ;HITS IN THE GROUP BEING
5256 027172 012737 027214 001510 MOV #1,$,LPERR ;TESTED.
5257 027200 013702 027526 MOV LLGS,R2
5258 027204 042702 177717 BIC #177717,R2
5259 027210 010237 177746 MOV R2,@#CONTRL
5260 027214 005011 1$: CLR (R1)
5261 027216 005711 TST (R1)
5262 027220 005721 TST (R1)+
5263 027222 032737 000010 177752 BIT #10,@#HITMIS
5264 027230 001011 BNE 3$
5265 027232 013737 027520 001634 MOV LLFLG1,$TMP1
5266 027240 011137 001636 MOV (R1),$TMP2
5267 027244 062737 177776 001636 ADD #-2,$TMP2
5268 027252 104001 2$: ERROR 1
5269 027254 077021 3$: SOB R0,1$
5270 027256 013700 027530 MOV LLGM,R0 ;FROM NOW ON SELECT
5271 027262 042700 177717 BIC #177717,R0 ;THE GROUP NOT BEING
5272 027266 010037 177746 MOV R0,@#CONTRL ;TESTED
5273
5274 027272 012701 140000 MOV #TESTR1,R1 ;INITIALIZE FOR TEST.
5275 027276 012700 001000 MOV #1000,R0 ;COUNTER.
5276 027302 005002 LL4: CLR R2 ;DATA PATTERN WRITTEN
    
```

```

5277 027304 005003          CLR      R3          ;LOGICAL 'OR' OF BAD DATA
5278 027306 012704 177777  MOV      #177777,R4  ;LOGICAL 'AND' OF BAD DATA
5279 027312 005005          CLR      R5          ;DATA PATTERN READ
5280 027314 005037 027532  CLR      LLCNT1     ;NUMBER OF LOCATIONS WHICH FAIL.
5281 027320 005037 027522  CLR      LLFLG2     ;ERROR IN GROUP FLAG
5282 027324 012737 027332 001510  MOV      #LL5,$LPERR
5283 027332 005037 027524  LL5:    CLR      LLFLG4     ;ERROR IN TESTED WORD FLAG.
5284 027336 000240          NOP                    ;FOR SCOPING WITH AN OSCILLOSCOPE.
5285 027340 010211          MOV      R2,(R1)
5286 027342 011105          MOV      (R1),R5
5287 027344 032737 000010 177752  BIT      #10,@#HITMIS
5288 027352 001006          BNE     LL6
5289 027354 013737 027520 001634  MOV      LLFLG1,$TMP1
5290 027362 010137 001636  MOV      R1,$TMP2
5291 027366 104001          1$:    ERROR      1
5292 027370 020205  LL6:    CMP      R2,R5     ;GOOD DATA
5293 027372 001402          BEQ     LL7
5294 027374 000137 027744  JMP      LLERR2     ;BAD DATA BUT NO TRAP OR
5295                                     ;ABORT OCCURRED!
5296 027400          LL7:    ;DECREMENT THE COUNT PATTERN
5297                                     ;AND LOOP IF NOT DONE
5298 027400 005737 027524  TST      LLFLG4     ;IF THERE WAS AN ERROR
5299 027404 001405          BEQ     LL8         ;IN THE WORD JUST TESTED
5300 027406 005237 027532  INC      LLCNT1     ;INCREMENT LLCNT1
5301 027412 012737 177777 027522  MOV      #-1,LLFLG2 ;AND SET ERROR IN GROUP FLAG.
5302 027420 062701 000002  LL8:    ADD      #2,R1     ;GO TO NEXT WORD.
5303 027424 077036          SOB     R0,LL5
5304
5305 027426 005737 027522  TST      LLFLG2     ;DONE WITH THAT GROUP,
5306 027432 001417          BEQ     LL9         ;SEE IF THERE WERE
5307 027434 112737 000013 001514  MOVB     #13,$ITEMB  ;ANY ERRORS. IF SO THEN
5308 027442 013737 027520 001634  MOV      LLFLG1,$TMP1 ;PRINT AN ERROR SUMMARY
5309 027450 010437 001636  MOV      R4,$TMP2   ;FOR THAT GROUP.
5310 027454 010337 001640  MOV      R3,$TMP3
5311 027460 013737 027532 001642  MOV      LLCNT1,$TMP4
5312 027466 004737 056354  JSR     PC,ERTYPE
5313
5314 027472 012737 000044 027530  LL9:    MOV      #S1M0,LLGM ;TEST THE OTHER GROUP, 0.
5315 027500 012737 000030 027526  MOV      #S0M1,LLGS ;
5316 027506 005337 027520  DEC      LLFLG1
5317 027512 001137          BNE     LL10       ;DONE?
5318 027514 000137 027106  JMP      LL2
5319
5320 027520 000000  LLFLG1: .WORD 0     ;GROUP BEING TESTED, 1 OR 0.
5321 027522 000000  LLFLG2: .WORD 0     ;ERROR OCCURRED IN GROUP FLAG.
5322
5323 027524 000000  LLFLG4: .WORD 0     ;ERROR OCCURRED IN WORD FLAG.
5324
5325 027526 000000  LLGS:   .WORD 0     ;PATTERNS FOR CONTROL REGISTER
5326 027530 000000  LLGM:   .WORD 0
5327
5328 027532 000000  LLCNT1: .WORD 0     ;GROUP ERROR COUNT
5329
5330 027534 000000  LLMER:  .WORD 0     ;TEMPORARY STORAGE FOR
5331                                     ;THE CACHE ERROR REGISTER.
5332 027536 000000  LLTMP1: .WORD 0
    
```

```

5333
5334 027540 013737 177744 027534 LLERR1: MOV @MEMERR,LLMER ;COME HERE ON PARITY
5335 027546 012737 004100 027536 MOV #4100,LLTMP1 ;ABORT OR TRAP.
5336 027554 005737 027520 TST LLFLG1 ;TESTING GROUP 1 OR 0?
5337 027560 001403 BEQ 1$
5338 027562 012737 004200 027536 MOV #4200,LLTMP1
5339 027570 023737 027536 027534 1$: CMP LLTMP1,LLMER ;WAS THE ERROR EXPECTED?
5340 027576 001402 BEQ 2$
5341 027600 000137 055440 JMP SPUR ;NO!
5342
5343 027604 020137 177740 2$: CMP R1,@LOADRS ;WAS THAT ADDRESS EXPECTED?
5344 027610 001402 BEQ 3$
5345 027612 000137 055440 JMP SPUR ;NO!
5346
5347 027616 012737 177777 027524 3$: MOV #-1,LLFLG4 ;SET WORD ERROR FLAG
5348 027624 050203 BIS R2,R3 ;DO 'OR' OF FAILING DATA
5349 027626 005102 COM R2
5350 027630 040204 BIC R2,R4 ;DO 'AND' OF FAILING DATA
5351 027632 005102 COM R2
5352 027634 011637 001634 MOV (SP),$TMP1
5353 027640 022626 CMP (SP)+,(SP)+
5354 027642 013737 027520 001636 MOV LLFLG1,$TMP2
5355 027650 010237 001640 MOV R2,$TMP3
5356 027654 010137 001650 MOV R1,$TMP7
5357 027660 013737 177740 001642 MOV @LOADRS,$TMP4
5358 027666 013737 177742 001644 MOV @HIADRS,$TMP5
5359 027674 042737 140000 001644 BIC #140000,$TMP5
5360 027702 013737 027534 001646 MOV LLMER,$TMP6
5361 027710 104011 ERROR 11 ;REPORT ERROR.
5362
5363 027712 012737 027724 000114 MOV #LLERR3,@CACHVEC ;BEFORE CONTINUING THE
5364 ;BAD PARITY IN THE WORD
5365 ;BEING TESTED MUST BE
5366 ;DEALT WITH!
5367 027720 005011 CLR (R1) ;THIS INSTRUCTION CLR (R1)
5368 027722 005711 TST (R1) ;SHOULD TRAP!
5369
5370 027724 012737 177777 177744 LLERR3: MOV #-1,@MEMERR ;CLR THE ERROR REGISTER
5371 027732 012737 027540 000114 MOV #LLERR1,@CACHVEC ;RESTORE THE PARITY ERROR
5372 027740 000137 027400 JMP LL7 ;VECTOR AND CONTINUE.
5373
5374 027744 012737 177777 027524 LLERR2: MOV #-1,LLFLG4 ;BAD DATA WAS READ BUT
5375 ;NO TRAP OR ABORT OCCURRED!
5376 027752 050203 BIS R2,R3 ;'OR' BAD DATA
5377 027754 005102 COM R2
5378 027756 040204 BIC R2,R4 ;'AND' BAD DATA
5379 027760 005102 COM R2
5380 027762 013737 027520 001634 MOV LLFLG1,$TMP1
5381 027770 010137 001640 MOV R1,$TMP3
5382 027774 010237 001642 MOV R2,$TMP4
5383 030000 010537 001644 MOV R5,$TMP5
5384
5385 030004 104012 1$: ERROR 12 ;REPORT ERROR.
5386
5387 030006 000137 027400 LL10: JMP LL7 ;CONTINUE TEST.
5388 030012
    
```

5389  
5390  
5391  
5392  
5393  
5394  
5395  
5396  
5397  
5398  
5399  
5400  
5401  
5402  
5403  
5404  
5405  
5406  
5407  
5408  
5409  
5410  
5411  
5412  
5413  
5414  
5415  
5416  
5417  
5418  
5419  
5420  
5421  
5422  
5423  
5424  
5425  
5426  
5427  
5428  
5429  
5430  
5431  
5432  
5433  
5434  
5435  
5436  
5437  
5438  
5439  
5440  
5441  
5442  
5443  
5444

```

*****
*TEST 30      CACHE DATA MEMORY PARITY CHECKERS LOW BYTE TEST
*
*THIS IS A TEST OF THE TWO CACHE DATA MEMORY PARITY
*CHECKERS FOR THE LOW BYTE, ONE FOR EACH GROUP. THE
*MAINTENANCE REGISTER IS USED TO FORCE A PARITY A
*PARITY ERROR AT EVERY DATA PATTERN WHICH HAS A ONE
*PARITY BIT. NOTE THAT THE CACHE DATA MEMORY PARITY HAS,
*EFFECTIVELY, ODD PARITY. THE MAINTENANCE FUNCTION ON THE
*CACHE DATA MEMORY PARITY CHECKERS HAS THE EFFECT OF
*FORCING THE PARITY BIT OF THE BYTE BEING CHECKED TO
*ZERO. THIS MEANS THAT ONCE THIS MAINTENANCE FUNCTION
*IS ENABLED THE ERROR WILL OCCUR ON A SUBSEQUENT
*READ OF A BYTE WITH A ONE PARITY BIT, THAT IS
*BYTES WITH ZERO PARITY BITS WILL NOT CAUSE THE ERROR.
*
*****
TST30:  SCOPE
MOV     #20,$TIMES      ;;DO 20 ITERATIONS
IIA=$TN
MOV     #TST31,SKAD    ;SET THE SKAD REGISTER
                        ;IN CASE THE TEST ABORTS.
MOVB   $TSTNM,$TMP0
MOV     #SPUR,@#CACHVEC
CLR     R0              ;THIS IS THE COUNTER CONTAINING
                        ;THE TEST DATA PATTERN
IIA1:   MOV     #IIA1,$LPERR
        JSR     PC,PARCNT ;SET IF THIS TEST PATTERN HAS
        BIT     #BIT0,R2  ;THE PARITY BIT SET (1), IF NOT
        BEQ    IIA2      ;GO TO THE NEXT PATTERN
        JMP    IIA7
IIA2:   MOV     #SOM1,@#CONTRL ;SELECT GROUP ZERO.
        MOV     #IIA1,@#CACHVEC ;SET UP FOR THE ERROR
        MOV     #IIA1,R5      ;MAKE THE TEST ADDRESS A
        TST    (R5)          ;HIT IN GROUP ZERO
        TST    (R5)          ;MAKE SURE IT IS A HIT
                        ;SEE IF REFERENCE ADDRESS
                        ;IS A HIT.
        BIT     #10,@#HITMIS
        BNE
                        ;IF NOT ERROR!
        MOV     R5,$TMP2
        MOV     #0,$TMP1
        ERROR  1
        SKIPT
                        ;ERROR FATAL. GO TO NEXT TEST.
1$:     MOV     #20,R4 ;THIS PATTERN WILL BE
        MOV     #MAINT,R2 ;PUT IN THE MAINTENANCE
        CLR    R1        ;REGISTER
        MOV     R0,(R5)  ;PUT THE TEST PATTERN IN
    
```

030012 000004  
030014 012737 000020 001702  
000031  
030022 012737 030464 055572  
030030 113737 001502 001632  
030036 012737 055440 000114  
030044 005000  
030046 012737 030046 001510  
030054 004737 056032  
030060 032702 000001  
030064 001402  
030066 000137 030444  
030072 012737 000030 177746  
030100 012737 030350 000114  
030106 012705 030346  
030112 005715  
030114 005715  
030116 032737 000010 177752  
030124 001007  
030126 010537 001636  
030132 012737 000000 001634  
030140 104001  
030142 104415  
030144 012704 000020  
030150 012702 177750  
030154 005001  
030156 010015

```

5445                                     ;THE TEST ADDRESS
5446 030160 000401                      BR      64$
5447                                     ;
5448                                     LOC=.      ;GET THE PC TO AN EVEN WORD BOUNDARY!!!
5449                                     LOC=-4&LOC
5450                                     LOC=LOC+4
5451                                     .=LOC
5452                                     ;
5453                                     ;THE REFERENCE TO THIS NEXT INSTRUCTION
5454                                     ;WILL MAKE THE COMPARE INSTRUCTION A HIT
5455                                     ;SO THAT NO SPURIOUS ERROR SHOULD OCCUR
5456                                     ;WHILE THE MAINTENANCE REGISTER IS SET!
5457 030164 010412                      64$:  MOV    R4,(R2)      ;TURN ON THE MAINT. REG.
5458 030166 021500                      CMP    (R5),R0      ;THE REFERENCE TO (R5)
5459 030170 010112                      MOV    R1,(R2)      ;SHOULD CAUSE THE ERROR.
5460                                     ;
5461 030172                                IIA3:
5462                                     ;THE ERROR DIDN'T OCCUR!
5463 030172 010037 001636                MOV    R0,$TMP2     ;REPORT FAILURE
5464 030176 012737 030346 001640        MOV    #IIAT1,$TMP3
5465 030204 005037 001642                CLR    $TMP4
5466 030210 104144                      64$:  ERROR 144
5467                                     ;
5468 030212 012737 030410 000114        IIA4:  MOV    #IIAR2,@#CACHVEC ;SET UP FOR THE GROUP ONE
5469 030220 012737 030212 001510        MOV    #IIA4,$LPERR ;ERROR
5470 030226 012737 000044 177746        MOV    #S1M0,@#CONTRL ;SELECT GROUP ONE
5471                                     ;
5472 030234 012705 030346                MOV    #IIAT1,R5    ;MAKE THE TEST ADDRESS A
5473 030240 005715                        TST    (R5)         ;HIT, IN GROUP ONE.
5474 030242 005715                        TST    (R5)
5475                                     ;
5476                                     ;SEE IF REFERENCE ADDRESS
5477 030244 032737 000010 177752        BIT    #10,@#HITMIS ;IS A HIT.
5478 030252 001007                        BNE    1$
5479                                     ;IF NOT ERROR!
5480 030254 010537 001636                MOV    R5,$TMP2
5481 030260 012737 000001 001634        MOV    #1,$TMP1
5482 030266 104001                        ERROR 1
5483                                     ;
5484 030270 104415                        SKIPT                ;ERROR FATAL. GO TO NEXT TEST.
5485                                     ;
5486                                     ;
5487 030272 012704 000100                1$:  MOV    #100,R4 ;THIS PATTERN WILL BE
5488 030276 012702 177750                MOV    #MAINT,R2    ;PUT IN THE MAINT. REG.
5489 030302 005001                        CLR    R1
5490 030304 010015                        MOV    R0,(R5)      ;PUT THE TEST PATTERN IN (R5),
5491                                     ;IIAT1.
5492 030306 000402                        BR     50$          ;PUT THE NEXT INSTRUCTION EXECUTED
5493                                     ;ON AN EVEN WORD BOUNDARY SO THE
5494                                     ;SUBSEQUENT INSTRUCTION, A CMP,
5495                                     ;WILL BE A HIT.
5496                                     ;
5497                                     LOC=.      ;GET THE PC TO AN EVEN WORD BOUNDARY!!!
5498                                     LOC=-4&LOC
5499                                     LOC=LOC+4
5500                                     .=LOC

```



```

5501
5502 030314 000240          50$:  NOP                ;FOR SCOPING WITH AN OSCILLOSCOPE.
5503 030316 010412          MOV      R4,(R2)         ;TURN ON THE MAINT. REG.
5504 030320 021500          CMP      (R5),R0        ;THIS REFERENCE TO (R5) SHOULD
5505 030322 010112          MOV      R1,(R2)         ;CAUSE THE ERROR.
5506
5507 030324          IIA5:
5508                                ;THE ERROR DIDN'T OCCUR!
5509 030324 010037 001636          MOV      R0,$TMP2       ;REPORT FAILURE
5510 030330 012737 030346 001640          MOV      #IIAT1,$TMP3
5511 030336 005037 001642          CLR      $TMP4
5512 030342 104145          64$:  ERROR      145
5513
5514 030344 000437          IIA6:  BR      IIA7
5515
5516 030346 000000          IIAT1:.WORD    0
5517
5518 030350          IIA7:
5519 030350 022737 004500 177744          CMP      #4500,@#MEMERR ;MAKE SURE THE ERROR
5520 030356 001402          BEQ      2$            ;REGISTER IS SET PROPERLY
5521 030360 000137 055440          1$:  JMP      SPUR
5522 030364 022737 030346 177740          2$:  CMP      #IIAT1,@#LOADRS ;MAKE SURE THE ERROR
5523 030372 001372          BNE      1$            ;OCCURRED AT THE CORRECT
5524                                ;ADDRESS.
5525 030374 022626          CMP      (SP)+,(SP)+    ;RESET THE STACK
5526 030376 012737 177777 177744          MOV      #-1,@#MEMERR  ;CLEAR THE ERROR REGISTERS.
5527 030404 000137 030212          JMP      IIA4          ;GO TEST GROUP ONE
5528 030410
5529 030410 022737 004600 177744          IIA2:  CMP      #4600,@#MEMERR ;MAKE SURE THE ERROR
5530 030416 001402          BEQ      2$            ;REGISTER IS SET PROPERLY
5531 030420 000137 055440          1$:  JMP      SPUR
5532 030424 022737 030346 177740          2$:  CMP      #IIAT1,@#LOADRS ;MAKE SURE THE ERROR
5533 030432 001372          BNE      1$            ;OCCURRED AT THE CORRECT
5534                                ;ADDRESS.
5535 030434 022626          CMP      (SP)+,(SP)+    ;RESET THE STACK
5536 030436 012737 177777 177744          MOV      #-1,@#MEMERR  ;CLEAR THE ERROR REGISTERS.
5537
5538 030444 022700 000377          IIA7:  CMP      #377,R0    ;INCREMENT THE TEST
5539 030450 001404          BEQ      IIA8          ;PATTERN
5540 030452 062700 000001          ADD      #1,R0
5541 030456 000137 030046          JMP      IIA1
5542
5543 030462 104414          IIA8:  RSET

```

```

5544
5545 ::*****
5546 :*TEST 31          CACHE DATA MEMORY PARITY CHECKERS HIGH BYTE TEST
5547 :*
5548 :*THIS IS A TEST OF THE TWO CACHE DATA MEMORY PARITY
5549 :*CHECKERS FOR THE HIGH BYTE, ONE FOR EACH GROUP. THE
5550 :*MAINTENANCE REGISTER ISUSED TO FORCE A PARITY A
5551 :*PARITY ERROR AT EVERY DATA PATTERN WHICH HAS A ONE
5552 :*PARITY BIT. NOTE THAT THE CACHE DATA MEMORY PARITY HAS,
5553 :*EFFECTIVELY, ODD PARITY. THE MAINTENANCE FUNCTION ON THE
5554 :*CACHE DATA MEMORY PARITY CHECKERS HAS THE EFFECT OF
5555 :*FORCING THE PARITY BIT OF THE BYTE BEING CHECKED TO
5556 :*ZERO. THIS MEANS THAT ONCE THIS MAINTENANCE FUNCTION

```

```

5557      ;*IS ENABLED THE ERROR WILL OCCUR ON A SUBSEQUENT
5558      ;*READ OF A BYTE WITH A ONE PARITY BIT, THAT IS
5559      ;*BYTES WITH ZERO PARITY BITS WILL NOT CAUSE THE ERROR.
5560      ;*
5561      ;*****
5562 030464 000004      TST31: SCOPE
5563 030466 012737 000020 001702      MOV      #20,$TIMES      ;;DO 20 ITERATIONS
5564      000032      IIB=$TN
5565      ;SET THE SKAD REGISTER
5566 030474 012737 031140 055572      MOV      #TST32,SKAD    ;IN CASE THE TEST ABORTS.
5567
5568 030502 113737 001502 001632      MOVB     $TSTNM,$TMP0
5569 030510 012737 055440 000114      MOV      #SPUR,@#CACHVEC
5570
5571 030516 005000      CLR      R0              ;THIS IS THE COUNTER CONTAINING
5572      ;THE TEST DATA PATTERN
5573 030520 012737 030520 001510      IIB1:  MOV      #IIB1,$LPERR
5574 030526 004737 056032      JSR      PC,PARCNT      ;SET IF THIS TEST PATTERN HAS
5575 030532 032702 000001      BIT      #BIT0,R2      ;THE PARITY BIT SET (1), IF NOT
5576 030536 001402      BEQ      IIB2          ;GO TO THE NEXT PATTERN
5577 030540 000137 031120      JMP      IIB7
5578 030544 012737 000030 177746      IIB2:  MOV      #SOM1,@#CONTRL ;SELECT GROUP ZERO.
5579 030552 012737 031024 000114      MOV      #IIBR1,@#CACHVEC ;SET UP FOR THE ERROR
5580 030560 012705 031022      MOV      #IIBT1,R5     ;MAKE THE TEST ADDRESS A
5581 030564 005715      TST      (R5)          ;HIT IN GROUP ZERO
5582 030566 005715      TST      (R5)          ;MAKE SURE IT IS A HIT
5583
5584      ;SEE IF REFERENCE ADDRESS
5585 030570 032737 000010 177752      BIT      #10,@#HITMIS  ;IS A HIT.
5586 030576 001007      BNE
5587      ;IF NOT ERROR!
5588 030600 010537 001636      MOV      R5,$TMP2
5589 030604 012737 000000 001634      MOV      #0,$TMP1
5590 030612 104001      ERROR   1
5591
5592 030614 104415      SKIPT      ;ERROR FATAL. GO TO NEXT TEST.
5593
5594
5595 030616 012704 000040      1$:  MOV      #40,R4      ;THIS PATTERN WILL BE
5596 030622 012702 177750      MOV      #MAINT,R2    ;PUT IN THE MAINTENANCE
5597 030626 005001      CLR      R1          ;REGISTER
5598 030630 010015      MOV      R0,(R5)     ;PUT THE TEST PATTERN IN
5599      ;THE TEST ADDRESS
5600 030632 000402      BR      64$
5601
5602      030634      LOC=.              ;GET THE PC TO AN EVEN WORD BOUNDARY!!!
5603      030634      LOC=-4&LOC
5604      030640      LOC=LOC+4
5605      030640      .=LOC
5606
5607      ;THE REFERENCE TO THIS NEXT INSTRUCTION
5608      ;WILL MAKE THE COMPARE INSTRUCTION A HIT
5609      ;SO THAT NO SPURIOUS ERROR SHOULD OCCUR
5610      ;WHILE THE MAINTENANCE REGISTER IS SET!
5611 030640 010412      64$:  MOV      R4,(R2)
5612 030642 021500      CMP      (R5),R0     ;TURN ON THE MAINT. REG.
                        ;THE REFERENCE TO (R5)
    
```

```

5613 030644 010112          MOV      R1,(R2)          ;SHOULD CAUSE THE ERROR.
5614
5615 030646          IIB3:
5616
5617 030646 010037 001636          MOV      R0,$TMP2          ;THE ERROR DIDN'T OCCUR!
5618 030652 012737 031022 001640          MOV      #IIBT1,$TMP3      ;REPORT FAILURE
5619 030660 005037 001642          CLR      $TMP4
5620 030664 104146          64$:      ERROR      146
5621
5622 030666 012737 031064 000114          IIB4:      MOV      #IIBR2,@#CACHVEC      ;SET UP FOR THE GROUP ONE
5623 030674 012737 030666 001510          MOV      #IIB4,$LPERR      ;ERROR
5624 030702 012737 000044 177746          MOV      #S1M0,@#CONTRL    ;SELECT GROUP ONE
5625
5626 030710 012705 031022          MOV      #IIBT1,R5          ;MAKE THE TEST ADDRESS A
5627 030714 005715          TST      (R5)              ;HIT, IN GROUP ONE.
5628 030716 005715          TST      (R5)
5629
5630
5631 030720 032737 000010 177752          BIT      #10,@#HITMIS      ;SEE IF REFERENCE ADDRESS
5632 030726 001007          BNE      1$                ;IS A HIT.
5633
5634 030730 010537 001636          MOV      R5,$TMP2          ;IF NOT ERROR!
5635 030734 012737 000001 001634          MOV      #1,$TMP1
5636 030742 104001          ERROR      1
5637
5638 030744 104415          SKIPT                    ;ERROR FATAL. GO TO NEXT TEST.
5639
5640
5641 030746 012704 000200          1$:      MOV      #200,R4 ;THIS PATTERN WILL BE
5642 030752 012702 177750          MOV      #MAINT,R2          ;PUT IN THE MAINT. REG.
5643 030756 005001          CLR      R1
5644 030760 010015          MOV      R0,(R5)          ;PUT THE TEST PATTERN IN (R5),
5645
5646 030762 000402          BR       50$              ;IIBT1.
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656 030770 000240          50$:      NOP
5657 030772 010412          MOV      R4,(R2)          ;FOR SCOPING WITH AN OSCILLOSCOPE.
5658 030774 021500          CMP      (R5),R0          ;TURN ON THE MAINT. REG.
5659 030776 010112          MOV      R1,(R2)          ;THIS REFERENCE TO (R5) SHOULD
5660
5661 031000          IIB5:
5662
5663 031000 010037 001636          MOV      R0,$TMP2          ;THE ERROR DIDN'T OCCUR!
5664 031004 012737 031022 001640          MOV      #IIBT1,$TMP3      ;REPORT FAILURE
5665 031012 005037 001642          CLR      $TMP4
5666 031016 104147          64$:      ERROR      147
5667
5668 031020 000437          IIB6:      BR       IIB7
    
```

```

5669
5670 031022 000000 IIBT1:.WORD 0
5671
5672 031024 IIBR1:
5673 031024 022737 004500 177744 CMP #4500,@#MEMERR ;MAKE SURE THE ERROR
5674 031032 001402 BEQ 2$ ;REGISTER IS SET PROPERLY
5675 031034 000137 055440 1$: JMP SPUR
5676 031040 022737 031022 177740 2$: CMP #IIBT1,@#LOADRS ;MAKE SURE THE ERROR
5677 031046 001372 BNE 1$ ;OCCURRED AT THE CORRECT
5678 ;ADDRESS.
5679 031050 022626 CMP (SP)+,(SP)+ ;RESET THE STACK
5680 031052 012737 177777 177744 MOV #-1,@#MEMERR ;CLEAR THE ERROR REGISTERS.
5681 031060 000137 030666 JMP IIB4 ;GO TEST GROUP ONE
5682 031064 IIBR2:
5683 031064 022737 004600 177744 CMP #4600,@#MEMERR ;MAKE SURE THE ERROR
5684 031072 001402 BEQ 2$ ;REGISTER IS SET PROPERLY
5685 031074 000137 055440 1$: JMP SPUR
5686 031100 022737 031022 177740 2$: CMP #IIBT1,@#LOADRS ;MAKE SURE THE ERROR
5687 031106 001372 BNE 1$ ;OCCURRED AT THE CORRECT
5688 ;ADDRESS.
5689 031110 022626 CMP (SP)+,(SP)+ ;RESET THE STACK
5690 031112 012737 177777 177744 MOV #-1,@#MEMERR ;CLEAR THE ERROR REGISTERS.
5691
5692 031120 022700 177400 IIB7: CMP #177400,R0 ;INCREMENT THE TEST
5693 031124 001404 BEQ IIB8 ;PATTERN
5694 031126 062700 000400 ADD #400,R0
5695 031132 000137 030520 JMP IIB1
5696
5697 031136 104414 IIB8: RSET
5698
5699
5700 ;*****
5701 ;*TEST 32 CACHE DATA MEMORY WORST CASE NOISE TEST
5702 ;*
5703 ;*THIS TEST DOES A GALLOPING 0'S AND 1'S OR PING PONG
5704 ;*TEST ON THE CACHE BIPOLAR DATA MEMORY.
5705
5706 ;*
5707 ;*****
5708 031140 000004 TST32: SCOPE
5709 ;SET THE SKAD REGISTER
5710 031142 012737 032276 055572 MOV #TST33,SKAD ;IN CASE THE TEST ABORTS.
5711
5712
5713 031150 012737 055440 000114 MOV #SPUR,@#CACHVEC
5714 031156 113737 001502 001632 MOVB $TSTNM,$TMPO ;SAVE TESTN FOR PRINT OUT.
5715
5716 031164 005037 031666 CLR QQPAT1 ;BACK ROUND PATTERN OF
5717 ;0'S FOR THE GALLOPING
5718 ;1'S TEST TO BE EXECUTED
5719 ;FIRST.
5720 031170 012737 000001 031662 MOV #1,QQFLG2 ;QQFLG=1 MEANS GALLOPING
5721 ;ONES TEST IN PROGRESS.
5722 ;QQFLG=0 MEANS GALLOPING
5723 ;ZEROS TEST IN PROGRESS.
5724 031176 012737 031176 001510 QQ1: MOV #QQ1,$LPERR ;SET ERROR LOOP INITIALLY
    
```

```

5725
5726 031204 012737 000044 031676      MOV      #S1M0,QQGS      :TO THIS POINT.
5727 031212 012737 000030 031700      MOV      #S0M1,QQGM      :TEST GROUP 1 FIRST.
5728
5729
5730
5731
5732
5733
5734 031220 012737 000001 031664      MOV      #1,QQFLG1      :S0M1 AND S1M0 ARE
5735
5736
5737
5738 031226 012703 031226      QQ2:    MOV      #QQ2,R3      :MAKE LOCATIONS QQ1
5739 031232 012704 001000      MOV      #1000,R4      :THROUGH QQ2 + 2000 (OCT)
5740 031236 013737 031700 177746 1$:    MOV      QQGM,@#CONTRL  :HITS IN THE GROUP NOT
5741 031244 005713      TST      (R3)          :BEING TESTED WHILE
5742 031246 013737 031676 177746      MOV      QQGS,@#CONTRL  :GETTING THESE LOCATIONS
5743 031254 005763 002000      TST      2000(R3)      :TO BE MISSES IN THE
5744 031260 062703 000002      ADD      #2,R3         :GROUP THAT IS BEING
5745 031264 077414      SOB      R4,1$        :TESTED
5746 031266 012704 001000      MOV      #1000,R4      :MAKE LOCATIONS TESTR2
5747 031272 012705 142000      MOV      #TESTR2,R5     :THROUGH TESTR2+2000(OCT)
5748 031276 013703 031676      MOV      QQGS,R3       :HITS IN THE GROUP
5749 031302 042703 177717      BIC      #177717,R3     :BEING TESTED WHILE
5750 031306 010337 177746      MOV      R3,@#CONTRL   :WRITING THE BACKGROUND
5751 031312 013715 031666      QQ3:    MOV      QQPAT1,(R5) :PATTERN, IN QQPAT1, IN
5752 031316 005715      TST      (R5)
5753 031320 005725      TST      (R5)+        :THEM. MAKE SURE THEY
5754 031322 032737 000010 177752      BIT      #10,@#HITMIS  :ARE HITS
5755 031330 001011      BNE      QQ4
5756 031332 013737 031664 001634      MOV      QQFLG1,$TMP1  :IF NOT ERROR
5757 031340 010537 001636      MOV      R5,$TMP2
5758 031344 062737 177776 001636      ADD      #-2,$TMP2
5759 031352 104001      1$:    ERROR 1
5760 031354 077422      QQ4:    SOB      R4,QQ3
5761 031356 013703 031700      MOV      QQGM,R3       :FROM NOW ON SELECT
5762 031362 042703 177717      BIC      #177717,R3     :THE GROUP NOT BEING
5763 031366 010337 177746      MOV      R3,@#CONTRL   :TESTED
5764
5765 031372 012704 031702      MOV      #QQ10,R4      :THE THREE ROUTINES
5766 031376 042704 176000      BIC      #176000,R4     :QQ10-QQ11, QQ12-QQ13 AND
5767 031402 012705 031756      MOV      #QQ11,R5      :QQ14-QQ15 ARE IDENTICAL
5768 031406 042705 176000      BIC      #176000,R5     :EXCEPT FOR WHAT PART
5769 031412 020405      CMP      R4,R5         :OF THE CACHE GROUP THAT
5770 031414 002407      BLT      QQ5           :IS NOT BEING TEST THEY
5771 031416 012737 031760 031656      MOV      #QQ12,QQLO    :LIE IN. HERE DECIDE
5772 031424 012737 032036 031660      MOV      #QQ14,QQHI    :WHICH TWO OF THE
5773 031432 000450      BR       QQ8           :ABOVE THREE IS APPROPRIATE
5774 031434 012704 031760      QQ5:    MOV      #QQ12,R4      :FOR THIS TEST.
5775 031440 042704 176000      BIC      #176000,R4
5776 031444 012705 032034      MOV      #QQ13,R5
5777 031450 042705 176000      BIC      #176000,R5
5778 031454 020405      CMP      R4,R5
5779 031456 002407      BLT      QQ6
5780 031460 012737 032036 031656      MOV      #QQ14,QQLO

```

```

5781 031466 012737 031702 031660      MOV      #QQ10,QQHI
5782 031474 000427                    BR       QQ8
5783 031476 013704 031702      QQ6:    MOV      QQ10,R4
5784 031502 042704 176000      BIC     #176000,R4
5785 031506 012705 031760      MOV     #QQ12,R5
5786 031512 042705 176000      BIC     #176000,R5
5787 031516 020405                    CMP     R4,R5
5788 031520 003007                    BGT     QQ7
5789 031522 012737 031702 031656      MOV     #QQ10,QQLO
5790 031530 012737 031760 031660      MOV     #QQ12,QQHI
5791 031536 000406                    BR      QQ8
5792 031540 012737 031760 031656      QQ7:    MOV     #QQ12,QQLO
5793 031546 012737 031702 031660      MOV     #QQ10,QQHI
5794
5795 031554 012702 142000      QQ8:    MOV     #TESTR2,R2      ;INITIALIZE FOR EITHER
5796 031560 012701 140000      MOV     #TESTR1,R1      ;THE GALLOPING ONES OR
5797 031564 012705 001000      MOV     #1000,R5        ;GALLOPING ZEROES TEST
5798                                     ;WHICH IS PENDING.
5799 031570 012737 032200 000114      MOV     #QQERR1,@#CACHEVEC ;IF THE TEST FAILS A
5800                                     ;PARITY ABORT IS LIKELY
5801                                     ;SO SET UP TO GO THE
5802                                     ;ERROR ROUTINE.
5803 031576 012737 031604 001510      MOV     #QQ9,$LPERR     ;SET THE LOOP ERROR
5804                                     ;ADDRESS FOR THE BEGINNING
5805                                     ;OF THE PASS ROUTINE.
5806
5807 031604 012703 142000      QQ9:    MOV     #TESTR2,R3      ;THIS DOES ONE PASS OF
5808 031610 012704 001000      MOV     #1000,R4        ;THE TEST FOR EACH LOCATION.
5809 031614 005112      COM     (R2)            ;PUT THE GALLOPING PATTERN
5810                                     ;IN THE MEMORY.
5811
5812 031616 010100      QQ9.5: MOV     R1,R0          ;SEE WHICH OF THE
5813 031620 042700 176000      BIC     #176000,R0      ;TWO ROUTINES (QQ10,QQ12 OR
5814 031624 013737 031660 031670      MOV     QQHI,QQTMP1     ;QQ14) SHOULD FINISH
5815 031632 042737 176000 031670      BIC     #176000,QQTMP1 ;SETTING FOR THIS TEST
5816 031640 020037 031670      CMP     R0,QQTMP1      ;PASS.
5817 031644 002402      BLT     1$
5818 031646 000177 000004      JMP     @QQLO
5819 031652 000177 000002      1$:    JMP     @QQHI
5820
5821 031656 000000      QQLO:  .WORD  0          ;QQLO AND QQHI CONTAIN THE
5822 031660 000000      QQHI:  .WORD  0          ;ADDRESSES OF THE ROUTINES
5823                                     ;TO BE USED IN SETTING UP
5824                                     ;FOR A PASS.
5825 031662 000000      QQFLG2: .WORD  0        ;1 IF DOING GALLOPING 1'S TEST.
5826                                     ;0 IF DOING GALLOPING 0'S TEST.
5827 031664 000000      QQFLG1: .WORD  0        ;GROUP BEING TESTED, 1 OR 0.
5828 031666 000000      QQPAT1: .WORD  0        ;0 OR 1 BACKGROUND PATTERN.
5829 031670 000000      QQTMP1: .WORD  0        ;USED AS TEMPORARY STORAGE.
5830 031672 000000      QQTMP2: .WORD  0
5831 031674 000000      QQTMP3: .WORD  0
5832 031676 000000      QQGS:  .WORD  0
5833 031700 000000      QQGM:  .WORD  0
5834                                     ;THESE REGISTERS HOLD PATTERNS
5835                                     ;WHICH ARE TO BE LOADED INTO THE
5836                                     ;CACHE CONTROL REGISTER.

```

;THIS ROUTINE IS USED TO SET UP THE INSTRUCTIONS:

```

5837      :      1$:      CMP      (R3)+,(R2)
5838      :      SOB      R4,1$
5839      :      JMP      @#QQ16
5840      :IN POSITION, AS HITS IN THE GROUP NOT BEING TESTED.
5841 031702 000240      QQ10:  NOP      ;USED AS A BUFFER SO
5842 031704 000240      NOP      ;THIS CODE WON'T WIPE
5843      :      OUT DESIRED HITS
5844 031706 012711 022312      MOV      #022312,(R1) ;020323=(CMP (R3)+,(R2)
5845 031712 005711      TST      (R1)
5846 031714 012761 077402 000002      MOV      #077402,2(R1) ;077402=(SOB R4,..-2)
5847 031722 005761 000002      TST      2(R1)
5848 031726 012761 000137 000004      MOV      #000137,4(R1) ;000137=(JMP @#QQ16)
5849 031734 005761 000004      TST      4(R1) ;QQ16
5850 031740 012761 032114 000006      MOV      #QQ16,6(R1)
5851 031746 005761 000006      TST      6(R1)
5852 031752 000111      JMP      (R1) ;GO DO A PASS.
5853 031754 000240      QQ11:  NOP
5854 031756 000240
5855
5856      :THIS ROUTINE IS USED TO SET UP THE INSTRUCTIONS:
5857      :      1$:      CMP      (R3)+,(R2)
5858      :      SOB      R4,1$
5859      :      JMP      @#QQ16
5860      :IN POSITION, AS HITS IN THE GROUP NOT BEING TESTED.
5861 031760 000240      QQ12:  NOP      ;USED AS A BUFFER SO
5862 031762 000240      NOP      ;THIS CODE WON'T WIPE
5863      :      OUT DESIRED HITS
5864 031764 012711 022312      MOV      #022312,(R1) ;020323=(CMP (R3)+,(R2)
5865 031770 005711      TST      (R1)
5866 031772 012761 077402 000002      MOV      #077402,2(R1) ;077402=(SOB R4,..-2)
5867 032000 005761 000002      TST      2(R1)
5868 032004 012761 000137 000004      MOV      #000137,4(R1) ;000137=(JMP @#QQ16)
5869 032012 005761 000004      TST      4(R1) ;QQ16
5870 032016 012761 032114 000006      MOV      #QQ16,6(R1)
5871 032024 005761 000006      TST      6(R1)
5872 032030 000111      JMP      (R1) ;GO DO A PASS.
5873 032032 000240      QQ13:  NOP
5874 032034 000240
5875
5876      :THIS ROUTINE IS USED TO SET UP THE INSTRUCTIONS:
5877      :      1$:      CMP      (R3)+,(R2)
5878      :      SOB      R4,1$
5879      :      JMP      @#QQ16
5880      :IN POSITION, AS HITS IN THE GROUP NOT BEING TESTED.
5881 032036 000240      QQ14:  NOP      ;USED AS A BUFFER SO
5882 032040 000240      NOP      ;THIS CODE WON'T WIPE
5883      :      OUT DESIRED HITS
5884 032042 012711 022312      MOV      #022312,(R1) ;020323=(CMP (R3)+,(R2)
5885 032046 005711      TST      (R1)
5886 032050 012761 077402 000002      MOV      #077402,2(R1) ;077402=(SOB R4,..-2)
5887 032056 005761 000002      TST      2(R1)
5888 032062 012761 000137 000004      MOV      #000137,4(R1) ;000137=(JMP @#QQ16)
5889 032070 005761 000004      TST      4(R1) ;QQ16
5890 032074 012761 032114 000006      MOV      #QQ16,6(R1)
5891 032102 005761 000006      TST      6(R1)
5892 032106 000111      JMP      (R1) ;GO DO A PASS.
    
```

```

5893 032110 000240      NOP
5894 032112 000240      QQ15:  NOP
5895
5896 032114 005122      QQ16:  COM      (R2)+      ;PASS DONE. RESTORE THE
5897                                     ;BACKGROUND PATTERN.
5898
5899 032116 062701 000002  QQ17:  ADD      #2,R1      ;GO TO NEXT LOCATION FOR
5900                                     ;NEXT PASS.
5901 032122 005305      DEC      R5      ;DO ANOTHER PASS?
5902 032124 001402      BEQ      1$
5903 032126 000137 031604      JMP      QQ9
5904 032132
5905 032132 012737 000044 031700  1$:  MOV      #S1M0,QQGM      ;TESTED GROUP 1 NOW GO BACK
5906 032140 012737 000030 031676      MOV      #SOM1,QQGS      ;AND TEST GROUP 0
5907 032146 005337 031664      DEC      QQFLG1
5908 032152 001002      BNE      QQ18
5909 032154 000137 031226      JMP      QQ2
5910
5911 032160 012737 177777 031666  QQ18:  MOV      #-1,QQPAT1      ;GALLOPING 1'S TEST IS
5912 032166 005337 031662      DEC      QQFLG2      ;COMPLETE, ON BOTH GROUPS,
5913 032172 001041      BNE      QQ19      ;SET THE BACKGROUND PATTERN
5914 032174 000137 031176      JMP      QQ1      ;FOR GALLOPING 0'S AND GO
5915                                     ;BACK TO PERFORM THIS TEST
5916                                     ;ON BOTH GROUPS.
5917
5918 032200 013737 177744 001634  QQERR1:  MOV      @MEMERR,$TMP1      ;COME HERE IF DURING THE
5919 032206 013737 177740 001636      MOV      @LOADRS,$TMP2      ;TEST A TRAP OR ABORT
5920 032214 013737 177742 001640      MOV      @HIADRS,$TMP3      ;OCCURRED TO CACHVEC
5921 032222 011637 001642      MOV      (SP),$TMP4
5922 032226 022626      CMP      (SP)+,(SP)+
5923 032230 010137 001644      MOV      R1,$TMP5
5924 032234 013737 031664 001646      MOV      QQFLG1,$TMP6
5925 032242 032737 000600 001634      BIT      #600,$TMP1
5926 032250 001002      BNE      QQERR2
5927 032252 104002      ERROR   2
5928 032254 000406      BR      QQERR4
5929 032256 005737 031666      QQERR2:  TST      QQPAT1      ;GALLOPING 1' OR 0'S?
5930 032262 001002      BNE      QQERR3
5931 032264 104003      ERROR   3      ;0'S.
5932 032266 000401      BR      QQERR4
5933 032270 104004      QQERR3:  ERROR   4      ;1'S
5934 032272 000137 032114      QQERR4:  JMP      QQ16      ;CONTINUE?
5935
5936 032276      QQ19:      ;DONE! PERHAPS PRINT SUMMARY.
5937                                     ;?????

```

```

5938
5939 ::*****
5940 ;*TEST 33      CACHE DATA MEMORY CHIP SELECTION LOGIC TEST
5941 ;*
5942 ;*THIS ROUTINE TESTS THE 'CHIP-SET' ENABLE LOGIC FOR THE CACHE DATA
5943 ;*MEMORY. TO DEFINE THE TERM 'CHIP-SET' CONSIDER THE CACHE MEMORY AS
5944 ;*BEING DIVIDED INTO FOUR SETS OF 256 (DEC) X 1 BIT BIPOLAR MEMORY
5945 ;*CHIPS. EACH SET IS MADE UP OF 18 CHIPS, THE 745200, EACH CHIP
5946 ;*REPRESENTS ONE BIT OF DATA OR PARITY, THUS 16 DATA BITS PLUS
5947 ;*TWO PARITY BITS CORRESPOND TO THE 18 CHIPS IN EACH GROUP.
5948 ;*THE 'CHIP-SETS' THEN CORRESPOND TO THE STRUCTURE OF THE MEMORY

```



5949  
5950  
5951  
5952  
5953  
5954  
5955  
5956  
5957  
5958  
5959  
5960  
5961  
5962  
5963  
5964  
5965  
5966  
5967  
5968  
5969  
5970  
5971  
5972  
5973  
5974  
5975  
5976  
5977  
5978  
5979  
5980  
5981  
5982  
5983  
5984  
5985  
5986  
5987  
5988  
5989  
5990  
5991  
5992  
5993  
5994  
5995  
5996  
5997  
5998  
5999  
6000  
6001  
6002  
6003  
6004

032276 000004  
032300 012737 000040 001702  
032306 012737 034042 055572  
032314 113737 001502 001632  
032322 012737 055440 000114  
032330 012737 000014 177746  
032336 005037 033676  
032342 012737 177777 033700  
032350 012737 125252 033702  
032356 012737 052525 033704  
032364 005037 033672  
032370 012700 033712  
032374 042700 176003  
032400 010001  
032402 062701 140000  
032406 010002  
032410 062702 142000  
032414 010137 001644  
032420 010137 001646  
032424 062737 000002 001646  
032432 010237 001650  
032436 010237 001652  
032442 062737 000002 001652  
032450 012705 033700  
032454 012700 000006  
032460 012737 000004 033674  
032466 012737 032474 001510

```

: *IN THIS WAY:
: *   SET 0   GROUP 0 EVEN WORD
: *   SET 1   GROUP 0 ODD WORD
: *   SET 2   GROUP 1 EVEN WORD
: *   SET 3   GROUP 1 ODD WORD
: *A DIFFERENT PATTERN, 000000 177777 125252 AND 052525, IS WRITTEN
: *INTO EACH GROUP AND THEN READ BACK. EVERY PERMUTATION OF THE
: *FOUR TEST PATTERNS IN THE FOUR SETS IS TRIED AND CHECKED.
: *FOR EACH PERMUTATION OF THE TEST PATTERNS THIS ROUTINE FIRST WRITES
: *'UP' (SET 0 FIRST THEN 1,2 AND 3) THEN 'DOWN' (SET 3 FIRST THEN 2,1 AND 0).
: *
: *****
TST33: SCOPE
MOV     #40,$TIMES      ;;DO 40 ITERATIONS
                ;;SET THE SKAD REGISTER
MOV     #TST34,SKAD    ;;IN CASE THE TEST ABORTS.

MOV     $TSTNM,$TMP0   ;PUT THE TEST NUMBER IN
                ;$TMP0 FOR PRINT OUT.
MOV     #SPUR,@#CACHVEC ;EXPECT NO PARITY ERRORS.

KK1:  MOV     #MOM1,@#CONTRL ;FORCE MISSES AND
        CLR     KKPAT1      ;INITIALIZE THE TEST PATTERN
        MOV     #177777,KKPAT2 ;TABLE
        MOV     #125252,KKPAT3
        MOV     #52525,KKPAT4

        CLR     KKFLG1      ;INITIALIZE KKFLG1:
                ;0 MEANS WRITE PATTERNS IN
                ;IN THE UPWARD DIRECTION
                ;1 MEANS WRITE PATTERNS IN
                ;THE DOWNWARD DIRECTION

KK2:  MOV     #KKTMP2,R0    ;ESTABLISH AN OFFSET FOR
        BIC     #176003,R0 ;A TEST AREA WHOSE HITS
                ;WILL NOT BE INTERFERRED WITH BY
        MOV     R0,R1      ;THE CYCLES CAUSED WHILE
        ADD     #TSTR1,R1  ;FETCHING THE TEST CODE.
        MOV     R0,R2
        ADD     #TSTR2,R2

        MOV     R1,$TMP5   ;SAVE THE ADDRESSES OF
        MOV     R1,$TMP6   ;THE FOUR TEST WORD LOCATIONS,
        ADD     #2,$TMP6   ;FOR TYPE OUT IN CASE
        MOV     R2,$TMP7   ;OF ERROR.
        MOV     R2,$TMP10
        ADD     #2,$TMP10

        MOV     #KKPAT2,R5 ;A POINTER USED IN GENERATING
                ;EVERY PERMUTATION OF THE TEST
                ;PATTERNS.
        MOV     #6,R0      ;R0 AND KKCNT1 ARE ALSO USED
        MOV     #4,KKCNT1 ;IN GENERATING THE PERMUTATIONS.

        MOV     #KK3,$LPERR ;WHEN LOOPING ON ERROR GO TO KK3.
    
```

```

6005 032474 000240          KK3:  NOP
6006 032476 012737 000034 177746  MOV  #SOMOM1,@#CONTRL ;FOR SCOPING PER POSES
6007 032504 005711          TST  (R1) ;MAKE THE TEST AREA HITS
6008 032506 005761 000002          TST  2(R1) ;IN THE CACHE GROUPS.
6009 032512 012737 000054 177746  MOV  #S1MOM1,@#CONTRL
6010 032520 005712          TST  (R2)
6011 032522 005762 000002          TST  2(R2)
6012 032526 005037 177746          CLR  @#CONTRL
6013
6014
6015 032532 005711          TST  (R1)
6016
6017
6018 032534 032737 000010 177752  BIT  #10,@#HITMIS ;SEE IF REFERENCE ADDRESS
6019 032542 001006          BNE  1$ ;IS A HIT.
6020
6021 032544 010137 001636          MOV  R1,$TMP2 ;IF NOT ERROR!
6022 032550 012737 000000 001634  MOV  #0,$TMP1
6023 032556 104001          ERROR 1
6024
6025
6026
6027 032560          1$:
6028
6029 032560 005761 000002          TST  2(R1)
6030
6031
6032 032564 032737 000010 177752  BIT  #10,@#HITMIS ;SEE IF REFERENCE ADDRESS
6033 032572 001011          BNE  2$ ;IS A HIT.
6034
6035 032574 010137 001636          MOV  R1,$TMP2 ;IF NOT ERROR!
6036 032600 062737 000002 001636  ADD  #2,$TMP2
6037 032606 012737 000000 001634  MOV  #0,$TMP1
6038 032614 104001          ERROR 1
6039
6040
6041
6042 032616          2$:
6043
6044 032616 005712          TST  (R2)
6045
6046
6047 032620 032737 000010 177752  BIT  #10,@#HITMIS ;SEE IF REFERENCE ADDRESS
6048 032626 001006          BNE  3$ ;IS A HIT.
6049
6050 032630 010237 001636          MOV  R2,$TMP2 ;IF NOT ERROR!
6051 032634 012737 000001 001634  MOV  #1,$TMP1
6052 032642 104001          ERROR 1
6053
6054
6055
6056 032644          3$:
6057
6058 032644 005762 000002          TST  2(R2)
6059
6060
    ;SEE IF REFERENCE ADDRESS
    
```

```

6061 032650 032737 000010 177752 BIT #10,@#HITMIS ;IS A HIT.
6062 032656 001011 BNE 4$ ;IF NOT ERROR!
6063
6064 032660 010237 001636 MOV R2,$TMP2
6065 032664 062737 000002 001636 ADD #2,$TMP2
6066 032672 012737 000001 001634 MOV #1,$TMP1
6067 032700 104001 ERROR 1
6068
6069
6070
6071
6072 032702 005737 033672 4$: TST KKFLG1 ;SEE IF THE TST PATTERN
6073 ;SHOULD BE WRITTEN UPWARD
6074 ;OR DOWNWARD.
6075 032706 001045 BNE KK4 ;BRANCH IF DOWNWARD
6076 ;OTHERWISE WRITE IT IN THE
6077 ;UPWARD DIRECTION.
6078 032710 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6079 032716 013703 033676 MOV KKPAT1,R3 ;LOCATION KKPAT1, INTO THE
6080 032722 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 0
6081 032726 010361 000000 MOV R3,0(R1)
6082 032732 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6083 032740 013703 033700 MOV KKPAT2,R3 ;LOCATION KKPAT2, INTO THE
6084 032744 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 2
6085 032750 010361 000002 MOV R3,2(R1)
6086 032754 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6087 032762 013703 033702 MOV KKPAT3,R3 ;LOCATION KKPAT3, INTO THE
6088 032766 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 0
6089 032772 010362 000000 MOV R3,0(R2)
6090 032776 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6091 033004 013703 033704 MOV KKPAT4,R3 ;LOCATION KKPAT4, INTO THE
6092 033010 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 2
6093 033014 010362 000002 MOV R3,2(R2)
6094 033020 000444 BR KK5
6095 033022 KK4: ;WRITE THE PATTERN IN THE
6096 ;DOWNWARD DIRECTION
6097 033022 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6098 033030 013703 033704 MOV KKPAT4,R3 ;LOCATION KKPAT4, INTO THE
6099 033034 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 2
6100 033040 010362 000002 MOV R3,2(R2)
6101 033044 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6102 033052 013703 033702 MOV KKPAT3,R3 ;LOCATION KKPAT3, INTO THE
6103 033056 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 0
6104 033062 010362 000000 MOV R3,0(R2)
6105 033066 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6106 033074 013703 033700 MOV KKPAT2,R3 ;LOCATION KKPAT2, INTO THE
6107 033100 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 2
6108 033104 010361 000002 MOV R3,2(R1)
6109 033110 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6110 033116 013703 033676 MOV KKPAT1,R3 ;LOCATION KKPAT1, INTO THE
6111 033122 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 0
6112 033126 010361 000000 MOV R3,0(R1)
6113
6114 033132 KK5:
6115 033132 012737 000014 177746 MOV #MOM1,@#CONTRL
6116 033140 013703 033676 MOV KKPAT1,R3 ;SEE IF THE TEST PATTERN WAS
    
```

6117	033144	005037	177746		CLR	@#CONTRL		;WRITTEN OR IS READ CORRECTLY.
6118	033150	016104	000000		MOV	0(R1),R4		
6119								
6120								;SEE IF REFERENCE ADDRESS
6121	033154	032737	000010	177752	BIT	#10,@#HITMIS		;IS A HIT.
6122	033162	001006			BNE	64\$		
6123								;IF NOT ERROR!
6124	033164	010137	001636		MOV	R1,\$TMP2		
6125	033170	012737	000000	001634	MOV	#0,\$TMP1		
6126	033176	104001			ERROR	1		
6127								
6128								
6129	033200	020403			64\$:	CMP	R4,R3	
6130	033202	001402				BEQ	KK6	
6131	033204	004737	033722			JSR	PC,KKERR1	
6132								
6133	033210				KK6:			
6134	033210	012737	000014	177746	MOV	#MOM1,@#CONTRL		
6135	033216	013703	033700		MOV	KKPAT2,R3		;SEE IF THE TEST PATTERN WAS
6136	033222	005037	177746		CLR	@#CONTRL		;WRITTEN OR IS READ CORRECTLY.
6137	033226	016104	000002		MOV	2(R1),R4		
6138								
6139								;SEE IF REFERENCE ADDRESS
6140	033232	032737	000010	177752	BIT	#10,@#HITMIS		;IS A HIT.
6141	033240	001011			BNE	64\$		
6142								;IF NOT ERROR!
6143	033242	010137	001636		MOV	R1,\$TMP2		
6144	033246	062737	000002	001636	ADD	#2,\$TMP2		
6145	033254	012737	000000	001634	MOV	#0,\$TMP1		
6146	033262	104001			ERROR	1		
6147								
6148								
6149	033264	020403			64\$:	CMP	R4,R3	
6150	033266	001402				BEQ	KK7	
6151	033270	004737	033734			JSR	PC,KKERR2	
6152								
6153	033274				KK7:			
6154	033274	012737	000014	177746	MOV	#MOM1,@#CONTRL		
6155	033302	013703	033702		MOV	KKPAT3,R3		;SEE IF THE TEST PATTERN WAS
6156	033306	005037	177746		CLR	@#CONTRL		;WRITTEN OR IS READ CORRECTLY.
6157	033312	016204	000000		MOV	0(R2),R4		
6158								
6159								;SEE IF REFERENCE ADDRESS
6160	033316	032737	000010	177752	BIT	#10,@#HITMIS		;IS A HIT.
6161	033324	001006			BNE	64\$		
6162								;IF NOT ERROR!
6163	033326	010237	001636		MOV	R2,\$TMP2		
6164	033332	012737	000001	001634	MOV	#1,\$TMP1		
6165	033340	104001			ERROR	1		
6166								
6167								
6168	033342	020403			64\$:	CMP	R4,R3	
6169	033344	001402				BEQ	KK8	
6170	033346	004737	033754			JSR	PC,KKERR3	
6171								
6172	033352				KK8:			

6173	033352	012737	000014	177746	MOV	#MOM1,@#CONTRL	
6174	033360	013703	033704		MOV	KKPAT4,R3	
6175	033364	005037	177746		CLR	@#CONTRL	;SEE IF THE TEST PATTERN WAS
6176	033370	016204	000002		MOV	2(R2),R4	;WRITTEN OR IS READ CORRECTLY.
6177							
6178							
6179	033374	032737	000010	177752	BIT	#10,@#HITMIS	;SEE IF REFERENCE ADDRESS
6180	033402	001011			BNE	64\$	;IS A HIT.
6181							;IF NOT ERROR!
6182	033404	010237	001636		MOV	R2,\$TMP2	
6183	033410	062737	000002	001636	ADD	#2,\$TMP2	
6184	033416	012737	000001	001634	MOV	#1,\$TMP1	
6185	033424	104001			ERROR	1	
6186							
6187							
6188	033426	020403			64\$: CMP	R4,R3	
6189	033430	001402			BEQ	KK10	
6190	033432	004737	033770		JSR	PC,KKERR4	
6191							
6192	033436	005737	033672		KK10: TST	KKFLG1	;SEE IF THIS PERMUTATION OF
6193	033442	001005			BNE	KK11	;THE TEST PATTERN HAS BEEN
6194	033444	012737	177777	033672	MOV	#-1,KKFLG1	;WRITTEN BOTH UPWARD AND
6195	033452	000137	032474		JMP	KK3	;DOWNWARD. IF NOT, KKFLG IS 0,
6196							;GO BACK TO WRITE IT DOWNWARD.
6197							
6198	033456	005037	033672		KK11: CLR	KKFLG1	;GENERATE THE NEXT PERMUTATION
6199	033462	012737	000014	177746	MOV	#MOM1,@#CONTRL	;OF THE TEST PATTERN IN THE
6200							;TEST TABLE
6201	033470	020527	033704		CMP	R5,#KKPAT4	
6202	033474	001011			BNE	KK12	
6203							
6204	033476	011537	033706		MOV	(R5),KKPAT5	
6205	033502	013715	033700		MOV	KKPAT2,(R5)	
6206	033506	012705	033700		MOV	#KKPAT2,R5	
6207	033512	013715	033706		MOV	KKPAT5,(R5)	
6208	033516	000406			BR	KK13	
6209							
6210	033520	012537	033706		KK12: MOV	(R5)+,KKPAT5	
6211	033524	011565	177776		MOV	(R5),-2(R5)	
6212	033530	013715	033706		MOV	KKPAT5,(R5)	
6213							
6214	033534	005300			KK13: DEC	R0	
6215	033536	001402			BEQ	KK14	
6216	033540	000137	032474		JMP	KK3	;GO DO NEXT PERMUTATION.
6217							
6218	033544	012700	000006		KK14: MOV	#6,R0	
6219	033550	013737	033676	033706	MOV	KKPAT1,KKPAT5	
6220	033556	005337	033674		DEC	KKCNT1	
6221							
6222	033562	022737	000003	033674	CMP	#3,KKCNT1	
6223	033570	001010			BNE	KK15	
6224							
6225	033572	013737	033700	033676	MOV	KKPAT2,KKPAT1	
6226	033600	013737	033706	033700	MOV	KKPAT5,KKPAT2	
6227	033606	000137	032474		JMP	KK3	;GO DO NEXT PERMUTATION.
6228							

```

6229 033612 022737 000002 033674 KK15:  CMP      #2,KKCNT1
6230 033620 001010                BNE      KK16
6231
6232 033622 013737 033702 033676      MOV      KKPAT3,KKPAT1
6233 033630 013737 033706 033702      MOV      KKPAT5,KKPAT3
6234 033636 000137 032474      JMP      KK3      ;GO DO NEXT PERMUTION.
6235
6236 033642 022737 000001 033674 KK16:  CMP      #1,KKCNT1
6237 033650 001073                BNE      KK17      ;BRANCH IF DONE!
6238
6239 033652 013737 033704 033676      MOV      KKPAT4,KKPAT1
6240 033660 013737 033706 033704      MOV      KKPAT5,KKPAT4
6241 033666 000137 032474      JMP      KK3      ;GO DO NEXT PERMUTATION.
6242
6243
6244 033672 000000                KKFLG1: .WORD 0      ;0 IF STORING PATTERN UPWARD
6245                                ;1 IF STORING DOWNWARD.
6246
6247 033674 000000                KKCNT1: .WORD 0      ;COUNTER USED IN GENERATING
6248                                ;THE TEST PATTERN PERMUTATIONS.
6249
6250 033676 000000                KKPAT1: .WORD 0      ;TEST PATTERN TABLE.
6251 033700 000000                KKPAT2: .WORD 0
6252 033702 000000                KKPAT3: .WORD 0
6253 033704 000000                KKPAT4: .WORD 0
6254 033706 000000                KKPAT5: .WORD 0
6255
6256 033710 000000                KKTMP1: .WORD 0      ;USED TO LOCATE A TEST AREA WHOSE
6257 033712 000000 000000 000000 KKTMP2: .WORD 0,0,0,0 ;HITS WON'T BE WIPED OUT BY TEST CODE.
6258 033720 000000
6259
6260 033722 010137 001642                KKERR1: MOV      R1,$TMP4      ;ERROR REPORTING ROUTINES
6261 033726 005037 001640                CLR      $TMP3
6262 033732 000427                BR       KKERR5
6263
6264 033734 010137 001642                KKERR2: MOV      R1,$TMP4
6265 033740 062737 000002 001642      ADD      #2,$TMP4
6266 033746 005037 001640                CLR      $TMP3
6267 033752 000417                BR       KKERR5
6268
6269 033754 010237 001642                KKERR3: MOV      R2,$TMP4
6270 033760 013737 000001 001640      MOV      1,$TMP3
6271 033766 000411                BR       KKERR5
6272
6273 033770 010237 001642                KKERR4: MOV      R2,$TMP4
6274 033774 062737 000002 001642      ADD      #2,$TMP4
6275 034002 012737 000001 001640      MOV      #1,$TMP3
6276 034010 000400                BR       KKERR5
6277
6278 034012 010337 001636                KKERR5: MOV      R3,$TMP2
6279 034016 011637 001634                MOV      (SP),$TMP1
6280 034022 012737 000014 177746      MOV      #MOM1,@#CONTRL
6281
6282 034030 104021                ERROR   21
6283
6284 034032 005037 177746                CLR      @#CONTRL
    
```

```

6285 034036 000207
6286
6287 034040 104414
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303 034042 000004
6304 034044 012737 000040 001702
6305
6306 034052 012737 035704 055572
6307
6308 034060 012737 055440 000114
6309 034066 113737 001502 001632
6310
6311
6312 034074 012737 001001 035542
6313 034102 012737 004004 035544
6314 034110 012737 020020 035546
6315 034116 012737 100100 035550
6316
6317 034124 012700 035554
6318 034130 042700 176003
6319 034134 010001
6320 034136 062701 140000
6321 034142 010002
6322 034144 062702 142000
6323
6324 034150 010137 001644
6325 034154 010137 001646
6326 034160 062737 000002 001646
6327 034166 010237 001650
6328 034172 010237 001652
6329 034176 062737 000002 001652
6330
6331 034204 012737 034212 001510
6332
6333 034212 000240
6334 034214 012737 000034 177746
6335 034222 005711
6336 034224 005761 000002
6337 034230 012737 000054 177746
6338 034236 005712
6339 034240 005762 000002
6340 034244 005037 177746

```

```

RTS PC
KK17: RSET ;DONE!
*****
*TEST 34 CACHE DATA MEMORY BYTE ENABLE LOGIC TEST
*
*THIS TEST PERFORMS A CHECK OF THE BYTE ENABLE LOGIC
*IN THE CACHE DATA MEMORY. THE BYTE PATTERNS 1, 2, 4, 10, 20,
*40, 100 A 200 ARE USED. THE FIRST FOUR PATTERNS ARE WRITTEN
*IN CONSECUTIVE BYTE LOCATIONS WHICH ARE HITS IN GROUP 0.
*THE REMAINING FOUR PATTERNS ARE WRITTEN IN CONSECUTIVE
*BYTE LOCATIONS WHICH ARE HITS IN GROUP 1. EACH PATTERN IS
*READ BACK CHECKED AND THE COMPLIMENT PATTERN IS WRITTEN.
*AFTER ALL THE PATTERNS HAVE BEEN CHECKED AND COMPLEMENTED
*THE COMPLIMENTED PATTERNS ARE CHECKED.
*
*****
TST34: SCOPE
MOV #40,$TIMES ;:DO 40 ITERATIONS
;SET THE SKAD REGISTER
MOV #TST35,SKAD ;:IN CASE THE TEST ABORTS.
MOV #SPUR,@#CACHVEC ;:ADDRESS AND PUT THE NO ERROR
MOVB $TSTNM,$TMP0 ;:EXPECTED ROUTINES ADDRESS IN
;THE PARITY ERROR VECTOR.
MM1: MOV #001001,MMPAT1 ;:SET UP THE PATTERN
MOV #004004,MMPAT2 ;:REGISTERS.
MOV #020020,MMPAT3
MOV #100100,MMPAT4
MOV #MMTMP2,R0 ;:LOCATE THE TEST AREA IN
BIC #176003,R0 ;:MEMORY WHOSE 'HITS' WILL NOT
MOV R0,R1 ;:INTERFER WITH HITS CAUSED
ADD #TSTR1,R1 ;:BY EXECUTING THIS TEST'S
MOV R0,R2 ;:CODE.
ADD #TSTR2,R2
MOV R1,$TMP5 ;:SAVE THE TEST AREA ADDRESSES
MOV R1,$TMP6 ;:FOR ERROR PRINT OUT.
ADD #2,$TMP6
MOV R2,$TMP7
MOV R2,$TMP10
ADD #2,$TMP10
MOV #MM2,$LPERR ;:SET THE LOOP ON ERROR REGISTER.
MM2: NOP
MOV #SOMOM1,@#CONTRL ;:MAKE THE TEST AREAS HITS
TST (R1) ;:IN GROUP 0 AND 1.
TST 2(R1)
MOV #S1MOM1,@#CONTRL
TST (R2)
TST 2(R2)
CLR @#CONTRL

```

6341							
6342							
6343	034250	005711				TST	(R1)
6344							
6345							
6346	034252	032737	000010	177752		BIT	#10,@#HITMIS
6347	034260	001006				BNE	MM3
6348							
6349	034262	010137	001636			MOV	R1,\$TMP2
6350	034266	012737	000000	001634		MOV	#0,\$TMP1
6351	034274	104001				ERROR	1
6352							
6353							
6354							
6355	034276						MM3:
6356							
6357	034276	005761	000002			TST	2(R1)
6358							
6359							
6360	034302	032737	000010	177752		BIT	#10,@#HITMIS
6361	034310	001011				BNE	MM4
6362							
6363	034312	010137	001636			MOV	R1,\$TMP2
6364	034316	062737	000002	001636		ADD	#2,\$TMP2
6365	034324	012737	000000	001634		MOV	#0,\$TMP1
6366	034332	104001				ERROR	1
6367							
6368							
6369							
6370	034334						MM4:
6371							
6372	034334	005712				TST	(R2)
6373							
6374							
6375	034336	032737	000010	177752		BIT	#10,@#HITMIS
6376	034344	001006				BNE	MM5
6377							
6378	034346	010237	001636			MOV	R2,\$TMP2
6379	034352	012737	000001	001634		MOV	#1,\$TMP1
6380	034360	104001				ERROR	1
6381							
6382							
6383							
6384	034362						MM5:
6385							
6386	034362	005762	000002			TST	2(R2)
6387							
6388							
6389	034366	032737	000010	177752		BIT	#10,@#HITMIS
6390	034374	001014				BNE	MM6
6391							
6392	034376	010237	001636			MOV	R2,\$TMP2
6393	034402	062737	000002	001636		ADD	#2,\$TMP2
6394	034410	012737	000001	001634		MOV	#1,\$TMP1
6395	034416	104001				ERROR	1
6396							

;SEE IF REFERENCE ADDRESS  
;IS A HIT.

;IF NOT ERROR!

;SEE IF REFERENCE ADDRESS  
;IS A HIT.

;IF NOT ERROR!

;SEE IF REFERENCE ADDRESS  
;IS A HIT.

;IF NOT ERROR!

;SEE IF REFERENCE ADDRESS  
;IS A HIT.

;IF NOT ERROR!



```

6397
6398
6399 034420 012737 034426 001510      MOV    #MM6,$LPERR      ;SET LOOP ON ERROR ADDRESS
6400 034426 012703 000001      MM6:  MOV    #1,R3
6401 034432 012704 000004      MOV    #4,R4
6402 034436 110321      MM7:  MOVVB  R3,(R1)+      ;PUT THE TEST PATTERN
6403 034440 006103      ROL    R3                ;IN GROUP 0
6404 034442 077403      SOB    R4,MM7
6405
6406 034444 012704 000004      MM8:  MOV    #4,R4
6407 034450 110322      MOVVB  R3,(R2)+      ;PUT THE TEST PATTERN
6408 034452 006103      ROL    R3                ;IN GROUP 1
6409 034454 077403      SOB    R4,MM8
6410 034456 010001      MOV    R0,R1
6411 034460 062701 140000      ADD    #TESTR1,R1      ;RE-ESTABLISH POINTERS TO
6412 034464 010002      MOV    R0,R2            ;THE TEST LOCATIONS.
6413 034466 062702 142000      ADD    #TESTR2,R2
6414 034472 012703 035542      MOV    #MMPAT1,R3      ;PUT THE ADDRESS OF THE TEST
6415                                     ;PATTERN REGISTERS IN R3
6416
6417 034476 005005      CLR    R5
6418
6419
6420 034500 005005      CLR    R5
6421 034502 111105      MOVVB  (R1),R5          ;GET THE PATTERN OUT OF
6422 034504 032737 000010 177752      BIT    #10,@#HITMIS    ;THIS BYTE MAKING SURE IT
6423 034512 001006      BNE    MM9              ;IS A HIT
6424 034514 010137 001636      MOV    R1,$TMP2
6425 034520 012737 000000 001634      MOV    #0,$TMP1
6426 034526 104001      ERROR  1
6427
6428 034530 042705 177400      MM9:  BIC    #177400,R5
6429 034534 022705 000001      CMP    #1,R5           ;SEE IF THE DATA IS CORRECT.
6430 034540 001402      BEQ    MM10
6431 034542 004737 035564      JSR    PC,MMERR1
6432 034546 105121      MM10: COMB   (R1)+      ;COMPLIMENT THE TEST PATTERN
6433 034550 012713 001376      MOV    #001376,(R3)
6434
6435
6436
6437 034554 005005      CLR    R5
6438 034556 111105      MOVVB  (R1),R5          ;GET THE PATTERN OUT OF
6439 034560 032737 000010 177752      BIT    #10,@#HITMIS    ;THIS BYTE MAKING SURE IT
6440 034566 001006      BNE    MM11              ;IS A HIT
6441 034570 010137 001636      MOV    R1,$TMP2
6442 034574 012737 000000 001634      MOV    #0,$TMP1
6443 034602 104001      ERROR  1
6444
6445 034604 042705 177400      MM11: BIC    #177400,R5
6446 034610 022705 000002      CMP    #2,R5           ;SEE IF THE DATA IS CORRECT.
6447 034614 001402      BEQ    MM12
6448 034616 004737 035564      JSR    PC,MMERR1
6449 034622 105121      MM12: COMB   (R1)+      ;COMPLIMENT THE TEST PATTERN
6450 034624 012713 176776      MOV    #176776,(R3)
6451
6452
    
```

6453	034630	062703	000002		ADD	#2,R3	:POINT TO THE NEXT ELEMENT :IN THE TEST PATTERN TABLE.
6454							
6455							
6456	034634	005005			CLR	R5	
6457	034636	111105			MOVB	(R1),R5	:GET THE PATTERN OUT OF
6458	034640	032737	000010	177752	BIT	#10,@#HITMIS	:THIS BYTE MAKING SURE IT
6459	034646	001006			BNE	MM13	:IS A HIT
6460	034650	010137	001636		MOV	R1,\$TMP2	
6461	034654	012737	000000	001634	MOV	#0,\$TMP1	
6462	034662	104001			ERROR	1	
6463							
6464	034664	042705	177400		MM13:	BIC	#177400,R5
6465	034670	022705	000004			CMP	#4,R5 ;SEE IF THE DATA IS CORRECT.
6466	034674	001402				BEQ	MM14
6467	034676	004737	035564			JSR	PC,MMERR1
6468	034702	105121			MM14:	COMB	(R1)+ ;COMPLIMENT THE TEST PATTERN
6469	034704	012713	004373			MOV	#004373,(R3)
6470							
6471							
6472							
6473	034710	005005			CLR	R5	
6474	034712	111105			MOVB	(R1),R5	:GET THE PATTERN OUT OF
6475	034714	032737	000010	177752	BIT	#10,@#HITMIS	:THIS BYTE MAKING SURE IT
6476	034722	001006			BNE	MM15	:IS A HIT
6477	034724	010137	001636		MOV	R1,\$TMP2	
6478	034730	012737	000000	001634	MOV	#0,\$TMP1	
6479	034736	104001			ERROR	1	
6480							
6481	034740	042705	177400		MM15:	BIC	#177400,R5
6482	034744	022705	000010			CMP	#10,R5 ;SEE IF THE DATA IS CORRECT.
6483	034750	001402				BEQ	MM16
6484	034752	004737	035564			JSR	PC,MMERR1
6485	034756	105121			MM16:	COMB	(R1)+ ;COMPLIMENT THE TEST PATTERN
6486	034760	012713	173773			MOV	#173773,(R3)
6487							
6488							
6489	034764	062703	000002		ADD	#2,R3	:POINT TO THE NEXT ELEMENT :IN THE TEST PATTERN TABLE.
6490							
6491							
6492	034770	005005			CLR	R5	
6493	034772	111205			MOVB	(R2),R5	:GET THE PATTERN OUT OF
6494	034774	032737	000010	177752	BIT	#10,@#HITMIS	:THIS BYTE MAKING SURE IT
6495	035002	001006			BNE	MM17	:IS A HIT
6496	035004	010237	001636		MOV	R2,\$TMP2	
6497	035010	012737	000001	001634	MOV	#1,\$TMP1	
6498	035016	104001			ERROR	1	
6499							
6500	035020	042705	177400		MM17:	BIC	#177400,R5
6501	035024	022705	000020			CMP	#20,R5 ;SEE IF THE DATA IS CORRECT.
6502	035030	001402				BEQ	MM18
6503	035032	004737	035576			JSR	PC,MMERR2
6504	035036	105122			MM18:	COMB	(R2)+ ;COMPLIMENT THE TEST PATTERN
6505	035040	012713	020357			MOV	#020357,(R3)
6506							
6507							
6508							

6509	035044	005005			CLR	R5	
6510	035046	111205			MOVB	(R2),R5	:GET THE PATTERN OUT OF
6511	035050	032737	000010	177752	BIT	#10,@#HITMIS	:THIS BYTE MAKING SURE IT
6512	035056	001006			BNE	MM19	:IS A HIT
6513	035060	010237	001636		MOV	R2,\$TMP2	
6514	035064	012737	000001	001634	MOV	#1,\$TMP1	
6515	035072	104001			ERROR	1	
6516							
6517	035074	042705	177400		MM19:	BIC	#177400,R5
6518	035100	022705	000040			CMP	#40,R5 ;SEE IF THE DATA IS CORRECT.
6519	035104	001402				BEQ	MM20
6520	035106	004737	035576			JSR	PC,MMERR2
6521	035112	105122			MM20:	COMB	(R2)+ ;COMPLIMENT THE TEST PATTERN
6522	035114	012713	157757			MOV	#157757,(R3)
6523							
6524							
6525	035120	062703	000002			ADD	#2,R3 ;POINT TO THE LAST ELEMENT
6526							:IN THE TEST PATTERN TABLE.
6527							
6528	035124	005005			CLR	R5	
6529	035126	111205			MOVB	(R2),R5	:GET THE PATTERN OUT OF
6530	035130	032737	000010	177752	BIT	#10,@#HITMIS	:THIS BYTE MAKING SURE IT
6531	035136	001006			BNE	MM21	:IS A HIT
6532	035140	010237	001636		MOV	R2,\$TMP2	
6533	035144	012737	000001	001634	MOV	#1,\$TMP1	
6534	035152	104001			ERROR	1	
6535							
6536	035154	042705	177400		MM21:	BIC	#177400,R5
6537	035160	022705	000100			CMP	#100,R5 ;SEE IF THE DATA IS CORRECT.
6538	035164	001402				BEQ	MM22
6539	035166	004737	035576			JSR	PC,MMERR2
6540	035172	105122			MM22:	COMB	(R2)+ ;COMPLIMENT THE TEST PATTERN
6541	035174	012713	100277			MOV	#100277,(R3)
6542							
6543							
6544							
6545	035200	005005			CLR	R5	
6546	035202	111205			MOVB	(R2),R5	:GET THE PATTERN OUT OF
6547	035204	032737	000010	177752	BIT	#10,@#HITMIS	:THIS BYTE MAKING SURE IT
6548	035212	001006			BNE	MM23	:IS A HIT
6549	035214	010237	001636		MOV	R2,\$TMP2	
6550	035220	012737	000001	001634	MOV	#1,\$TMP1	
6551	035226	104001			ERROR	1	
6552							
6553	035230	042705	177400		MM23:	BIC	#177400,R5
6554	035234	022705	000200			CMP	#200,R5 ;SEE IF THE DATA IS CORRECT.
6555	035240	001402				BEQ	MM24
6556	035242	004737	035576			JSR	PC,MMERR2
6557	035246	105122			MM24:	COMB	(R2)+ ;COMPLIMENT THE TEST PATTERN
6558	035250	012713	077677			MOV	#077677,(R3)
6559							
6560							
6561	035254	010001			MOV	R0,R1	:RE-ESTABLISH POINTERS TO
6562	035256	062701	140000		ADD	#TESTR1,R1	:THE TEST AREA
6563	035262	010002			MOV	R0,R2	
6564	035264	062702	142000		ADD	#TESTR2,R2	

```

6565
6566
6567 035270 012105          MOV    (R1)+,R5          ;CHECK THE COMPLIMENTED
6568
6569 035272 005761 177776    TST    -2(R1)
6570
6571
6572 035276 032737 000010 177752  BIT    #10,@#HITMIS    ;SEE IF REFERENCE ADDRESS
6573 035304 001011          BNE    MM25            ;IS A HIT.
6574
6575 035306 010137 001636          MOV    R1,$TMP2        ;IF NOT ERROR!
6576 035312 062737 177776 001636    ADD    #-2,$TMP2
6577 035320 012737 000000 001634    MOV    #0,$TMP1
6578 035326 104001          ERROR  1
6579
6580
6581
6582
6583 035330 020537 035542          MM25:  CMP    R5,MMPAT1          ;IS PATTERN CORRECT?
6584 035334 001402          BEQ    MM26
6585 035336 004737 035626          JSR    PC,MMERR4
6586
6587
6588 035342          MM26:
6589
6590 035342 012105          MOV    (R1)+,R5          ;CHECK THE COMPLIMENTED
6591
6592 035344 005761 177776    TST    -2(R1)
6593
6594
6595 035350 032737 000010 177752  BIT    #10,@#HITMIS    ;SEE IF REFERENCE ADDRESS
6596 035356 001011          BNE    MM27            ;IS A HIT.
6597
6598 035360 010137 001636          MOV    R1,$TMP2        ;IF NOT ERROR!
6599 035364 062737 177776 001636    ADD    #-2,$TMP2
6600 035372 012737 000000 001634    MOV    #0,$TMP1
6601 035400 104001          ERROR  1
6602
6603
6604
6605
6606 035402 020537 035544          MM27:  CMP    R5,MMPAT2          ;IS PATTERN CORRECT?
6607 035406 001402          BEQ    MM28
6608 035410 004737 035626          JSR    PC,MMERR4
6609
6610
6611 035414          MM28:
6612
6613 035414 012205          MOV    (R2)+,R5          ;CHECK THE COMPLIMENTED
6614
6615 035416 005762 177776    TST    -2(R2)
6616
6617
6618 035422 032737 000010 177752  BIT    #10,@#HITMIS    ;SEE IF REFERENCE ADDRESS
6619 035430 001011          BNE    MM29            ;IS A HIT.
6620
6620
6620
    
```

```

6621 035432 010237 001636      MOV      R2,$TMP2
6622 035436 062737 177776 001636      ADD      #-2,$TMP2
6623 035444 012737 000001 001634      MOV      #1,$TMP1
6624 035452 104001                ERROR    1
6625
6626
6627
6628
6629 035454 020537 035546      MM29:   CMP      R5,MMPAT3                ;IS PATTERN CORRECT?
6630 035460 001402                BEQ      MM30
6631 035462 004737 035646      JSR      PC,MMERR5
6632
6633
6634 035466                MM30:
6635
6636 035466 012205                MOV      (R2)+,R5                ;CHECK THE COMPLIMENTED
6637
6638 035470 005762 177776                TST      -2(R2)
6639
6640
6641 035474 032737 000010 177752      BIT      #10,@#HITMIS           ;SEE IF REFERENCE ADDRESS
6642 035502 001011                BNE      MM31                   ;IS A HIT.
6643
6644 035504 010237 001636      MOV      R2,$TMP2                ;IF NOT ERROR!
6645 035510 062737 177776 001636      ADD      #-2,$TMP2
6646 035516 012737 000001 001634      MOV      #1,$TMP1
6647 035524 104001                ERROR    1
6648
6649
6650
6651
6652 035526 020537 035550      MM31:   CMP      R5,MMPAT4           ;IS PATTERN CORRECT?
6653 035532 001464                BEQ      MM32
6654 035534 004737 035646      JSR      PC,MMERR5
6655
6656 035540 000461                BR       MM32                   ;FINISHED THIS TEST.
6657
6658 035542 000000      MMPAT1: .WORD    0                ;THIS IS THE TEST PATTERN
6659 035544 000000      MMPAT2: .WORD    0                ;TABLE.
6660 035546 000000      MMPAT3: .WORD    0
6661 035550 000000      MMPAT4: .WORD    0
6662
6663 035552 000000      MMTMP1: .WORD    0                ;THIS AREA IS USED TO ESTABLISH
6664 035554 000004      MMTMP2: .BLKW   4                ;A TEST LOCATION WHOSE HITS WON'T
6665
6666
6667
6668 035564 005037 001634      MMERR1: CLR      $TMP1           ;COME HERE TO REPORT
6669 035570 010137 001642      MOV      R1,$TMP4               ;GROUP 0 ERROR, WHILE READING
6670 035574 000405                BR       MMERR3                 ;A BYTE INTO R5
6671
6672 035576 012737 000001 001634      MMERR2: MOV      #1,$TMP1           ;COME HERE TO REPORT
6673 035604 010237 001642      MOV      R2,$TMP4               ;GROUP 1 ERROR, READING A
6674
6675 035610 012637 001636      MMERR3: MOV      (SP)+,$TMP2       ;BYTE INTO R5.
6676 035614 010537 001640      MOV      R5,$TMP3

```

```

6677
6678 035620 104017          ERROR 17
6679 035622 000177 144010    JMP @STMP2
6680
6681 035626 005037 001634    MMERR4: CLR $TMP1          ;REPORT AN ERROR IN GROUP
6682 035632 010137 001642    MOV R1,$TMP4          ;0 WHILE READING A WORD
6683 035636 062737 177776 001642  ADD #-2,$TMP4
6684 035644 000410          BR MMERR6
6685
6686 035646 012737 000001 001634  MMERR5: MOV #1,$TMP1
6687 035654 010237 001642    MOV R2,$TMP4
6688 035660 062737 177776 001642  ADD #-2,$TMP4
6689
6690 035666 012637 001636    MMERR6: MOV (SP)+,$TMP2
6691 035672 010537 001640    MOV R5,$TMP3
6692
6693 035676 104020          ERROR 20
6694 035700 000177 143732    JMP @STMP2
6695
6696 035704          MM32:          ;DONE!
6697
6698
6699
6700
6701
6702
6703
6704
6705
6706
6707
6708
6709
6710
6711
6712
6713
6714
6715
6716
6717
6718
6719
6720
6721
6722
6723
6724
6725
6726
6727
6728
6729
6730
6731
6732
    
```

```

:*****
:*TEST 35          CACHE ARBITRATION AND HIGH SPEED I/O TEST
:*
:*THIS IS A TEST OF:
:*  1.          CACHE ARBITRATION
:*  2.          THE MASS BUS AND UNIBUS PORTS TO THE CACHE
:*  3.          HIGH SPEED I/O THROUGH THE CACHE
:*
:*IT MAKE USE OF THE FOLLOWING DEVICES:
:*  RS04
:*  RP04
:*  RK05
:*  MASS BUSS TESTER
:*  UNIBUS EXERCISER
:*
:*IF ANY OF THESE DEVICES ARE PRESENT AND WRITE ENABLED THE WILL BE USED
:*IN THIS TEST. ONLY THE LOWEST WRITE ENABLED DRIVE NUMBER OF EACH DEVICE
:*WILL BE USED.
:*
:*  CAUTION!!!
:*  THIS TEST WILL WRITE ON THE DISKS IT USES. SO VITAL SYSTEMS
:*  DISKS SHOULD BE REMOVED OR WRITE PROTECTED BEFORE RUNNING
:*  THIS DIAGNOSTIC.
:*
:*IF UNIT ZERO OF A PARTICULAR DEVICE IS WRITE PROTECTED THEN THIS TEST
:*WILL TRY TO USE UNIT ONE, ETC.
:*
:*ALL AVAILABLE DEVICES ARE STARTED DOING TRANSFERS AT THE SAME TIME
:*TO DIFFERENT PARTS OF MEMORY.
:*EACH DEVICE HAS A CONTROL ROUTINE WHICH DRIVES THAT DEVICE THROUGH
:*THE CYCLE:
:*  1.          WRITE A RANDOM DATA PATTERN IN MEMORY
    
```

6733  
6734  
6735  
6736  
6737  
6738  
6739  
6740  
6741  
6742  
6743  
6744  
6745  
6746  
6747  
6748  
6749  
6750  
6751  
6752  
6753  
6754  
6755  
6756  
6757  
6758  
6759  
6760  
6761  
6762  
6763  
6764  
6765  
6766  
6767  
6768  
6769  
6770  
6771  
6772  
6773  
6774  
6775  
6776  
6777  
6778  
6779  
6780  
6781  
6782  
6783  
6784  
6785  
6786  
6787  
6788

```

: *      2.      COPY THAT PATTERN ONTO THE DISK
: *      3.      WRITE CHECK THE DISK
: *      4.      READ THE PATTERN OFF THE DISK BACK INTO MEMORY
: *      5.      CHECK DATA
: *      6.      START OVER AT 1.
: *
: *EACH DEVICE IS CAUSED TO GO THROUGH THIS CYCLE A PREDETERMINED
: *NUMBER OF TIMES. THIS NUMBER IS CONTAINED IN THE LOCATION,
: *CYCNT, AND CAN BE CHANGED BY THE USER AT THE CONSOLE TO ANY VALUE
: *HE DESIRES).
: *
: *INTERRUPTS ARE ENABLED SO THAT IT IS POSSIBLE TO GET MANY DEVICES
: *DOING TRANSFERS AT ONCE.
: *
: *UNFORTUNATELY THE DEGREE TO WHICH FAULTS CAN BE ISOLATED IS
: *LIMITED BY THE FACT THAT THERE ARE MANY ELEMENTS, DEVICES, INVOLVED.
: *THESE ERRORS ARE REPORTED:
: *      1.      ALL DEVICE ERRORS
: *      2.      ALL DATA OR PARITY ERRORS
: *
: *NOTE THAT THIS NOT INTENDED TO BE USED AS AN I/O DEVICE DIAGNOSTIC!
: *ALL THE DEVICES WHICH ARE USED ARE ASSUMED TO BE IN PROPER WORKING
: *CONDITION.
: *
: *
: *****

```

```

035704 000004
035706 012737 042352 055572
035714 104414
035716 113737 001502 001632
035724 012700 172340
035730 012701 077406
035734 012702 172300
035740 012703 000010
035744 010122
035746 077302
035750 005020
035752 012720 000200
035756 012720 000400
035762 012720 000600
035766 012720 001000
035772 012720 001200
035776 012720 001400
036002 012710 177600
036006 012737 000001 177572
036014 012737 000060 172516
036022 004737 042116
036026 004737 057066
036032 005046

```

```

TST35: SCOPE
MOV #TST36,SKAD ;SET THE SKAD REGISTER
;IN CASE THE TEST ABORTS.
RSET
MOVB $TSTNM,$TMPO
MOV #KIPAR0,R0 ;INITIALLY PUT MEMORY
MOV #77406,R1 ;MANAGEMENT IN A 'PASSIVE'
MOV #KIPDR0,R2 ;STATE, THAT IS MAP ALL
MOV #10,R3 ;VIRTUAL ADDRESSES ON TO
64$: MOV R1,(R2)+ ;THEMSELVES AS PHYSICAL
SOB R3,64$ ;ADDRESSES.
CLR (R0)+
MOV #200,(R0)+
MOV #400,(R0)+
MOV #600,(R0)+
MOV #1000,(R0)+
MOV #1200,(R0)+
MOV #1400,(R0)+
MOV #177600,(R0)
MOV #1,@MMR0
MOV #60,@MMR3
INT0: JSR PC,GTBINT ;INITIALIZE THE MEMORY BUFFER
;ALLOCATION ROUTINES.
JSR PC,SIZDEV ;GO DETERMINE WHAT DEVICES ARE
;PRESENT.
CLR -(SP) ;MAKE THE WAIT LOOP ACCESSABLE

```

```

6789 036034 012746 036304          MOV    #WAITLP,-(SP)          ;TO AN 'RTI'.
6790
6791 036040 012700 057460          INT1:  MOV    #RS4DFL,R0          ;GET READY TO SEE WHAT DEVICES
6792 036044 012701 036234          MOV    #RS4CR,R1             ;ARE TO BE USED.
6793 036050 012702 036246          MOV    #RS4SUN,R2
6794 036054 012703 036260          MOV    #RS4ASS,R3
6795 036060 012704 000005          MOV    #5,R4
6796
6797 036064 005011          INT2:  CLR    (R1)              ;CLEAR THE UNIT NUMBER.
6798 036066 005012          CLR    (R2)              ;CLEAR THE COUNTER.
6799 036070 105710          TSTB   (R0)              ;IS THERE A DRIVE.
6800 036072 001447          BEQ    INT6              ;BRANCH IF NOT.
6801
6802 036074 111005          MOVB   (R0),R5           ;OTHERWISE DETERMINE A UNIT NUM.
6803 036076 104412          SAVREG
6804 036100 012700 000010          MOV    #10,R0
6805 036104 005001          CLR    R1
6806 036106 012702 000001          MOV    #1,R2
6807 036112 030205          INT3:  BIT    R2,R5
6808 036114 001405          BEQ    INT4
6809 036116 010137 036230          MOV    R1,INTMP1
6810 036122 104413          RESREG
6811 036124 000137 036144          JMP    INT5
6812 036130 005201          INT4:  INC    R1
6813 036132 006302          ASL    R2
6814 036134 077012          SOB    R0,INT3
6815 036136 104413          RESREG
6816 036140 000137 036212          JMP    INT6
6817
6818 036144 013711 036232          INT5:  MOV    CYCNT,(R1)        ;FOUND THE DRIVE SO SET UP THE
6819 036150 020127 036234          CMP    R1,#RS4CR
6820 036154 001001          BNE    1$
6821 036156 006311          ASL    (R1)
6822 036160 020127 036236          1$:   CMP    R1,#RP4CR
6823 036164 001001          BNE    2$
6824 036166 006311          ASL    (R1)
6825 036170 020127 036240          2$:   CMP    R1,#RH4CR
6826 036174 001001          BNE    3$
6827 036176 006311          ASL    (R1)
6828 036200 012746 000340          3$:   MOV    #340,-(SP)        ;PASS COUNT AND MAKE THE DRIVER
6829 036204 011346          MOV    (R3),-(SP)        ;ACCESSIBLE BY A 'RTI'.
6830 036206 013712 036230          MOV    INTMP1,(R2)
6831
6832 036212 005200          INT6:  INC    R0
6833 036214 005721          TST    (R1)+             ;MOVE THE POINTERS TO THE NEXT DEVICE.
6834 036216 022223          CMP    (R2)+,(R3)+
6835 036220 000240          NOP
6836 036222 077460          SOB    R4,INT2
6837
6838
6839 036224 000240          NOP
6840 036226 000002          RTI                     ;START THE TEST!
6841
6842
6843
6844 036230 000000          ;THESE ARE SOME TABLES THAT ARE USED TO CONTROL AND SET UP THIS TEST.
          INTMP1: .WORD 0

```



```

6845
6846
6847 036232 000010      CYCNT:  .WORD  10      ;THE PASS COUNT!!!!
6848
6849 036234 000000      RS4CR:  .WORD  0      ;PASS COUNT FOR EACH DEVICE.
6850 036236 000000      RP4CR:  .WORD  0
6851 036240 000000      RH4CR:  .WORD  0
6852 036242 000000      RK5CR:  .WORD  0
6853 036244 000000      UBECR:  .WORD  0
6854
6855 036246 000000      RS4SUN: .WORD  0      ;THE DRIVE NUMBER USED FOR EACH
6856 036250 000000      RP4SUN: .WORD  0      ;DEVICE.
6857 036252 000000      RH4SUN: .WORD  0
6858 036254 000000      RK5SUN: .WORD  0
6859 036256 000000      UBESUN: .WORD  0
6860
6861          036260      SETBLE=RS4ASS
6862 036260 036350      RS4ASS: .WORD  DRRS4      ;STARTING ADDRESSES OF EACH DRIVER.
6863 036262 037162      RP4ASS: .WORD  DRRP4
6864 036264 037774      RH4ASS: .WORD  DRRH4
6865 036266 040566      RK5ASS: .WORD  DRRK5
6866 036270 041400      UBEASS: .WORD  DRUBE
6867
6868 036272 000000      RS4RB:  .WORD  0      ;WRITE AND READ BUFFERS OF EACH DEVICE.
6869 036274 000000      RP4RB:  .WORD  0
6870 036276 000000      RH4RB:  .WORD  0
6871 036300 000000      RK5RB:  .WORD  0
6872 036302 000000      UBERB:  .WORD  0
6873
6874
6875      ;THIS IS THE WAIT ROUTINE. COME HERE WHEN WAITING FOR AN INTERRUPT
6876      ;OR WHEN DONE, ALL THE PASS COUNTS HAVE GONE TO ZERO.
6877 036304 000230      WAITLP: SPL  0      ;LOWER THE PRIORITY.
6878 036306 005737 036242      TST  RK5CR      ;WAIT FOR INTERRUPT OR ZERO PASS COUNT.
6879 036312 001374      BNE  WAITLP
6880 036314 005737 036244      TST  UBECR
6881 036320 001371      BNE  WAITLP
6882 036322 005737 036236      TST  RP4CR
6883 036326 001366      BNE  WAITLP
6884 036330 005737 036234      TST  RS4CR
6885 036334 001363      BNE  WAITLP
6886 036336 005737 036240      TST  RH4CR
6887 036342 001360      BNE  WAITLP
6888
6889 036344 000137 042350      JMP  INDONE      ;FINISHED!!!
6890
6891
6892
6893      ;THIS IS THE RS4 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
6894      ;TEST.
6895
6896 036350 000240      DRRS4:  NOP
6897 036352 012737 007007 037156      MOV  #7007,DRS4T1      ;INITIALIZE THE RANDOM DISK ADDRESS
6898 036360 012737 006006 037160      MOV  #6006,DRS4T2      ;GENERATER.
6899 036366 012737 005005 036632      MOV  #5005,RS4AA3
6900
    
```

```

6901 036374 000240          RS4AA:  NOP
6902 036376 000240          NOP
6903 036400 104412          SAVREG
6904 036402 004737 042232  JSR     PC,GETBUF      ;GET A MEMORY BUFFER.
6905 036406 036272          .WORD  RS4RB
6906 036410 013701 036272  MOV     RS4RB,R1
6907 036414 005000          CLR     R0
6908 036416 073027 000014  ASHC   #12.,R0
6909
6910 036422 000237          SPL     7              ;GET A RANDOM DISK ADDRESS.
6911 036424 013737 037156 054370  MOV     DRS4T1,$HINUM
6912 036432 013737 037160 054372  MOV     DRS4T2,$LONUM
6913 036440 004737 054272          JSR     PC,$RAND
6914 036444 013737 054370 037156  MOV     $HINUM,DRS4T1
6915 036452 013737 054372 037160  MOV     $LONUM,DRS4T2
6916 036460 000230          SPL     0
6917
6918 036462 013702 036246          MOV     RS4SUN,R2      ;SET UP THE DEVICE UNIT NUM.
6919 036466 110237 037027          MOV     R2,RS4112
6920 036472 110237 036655          MOV     R2,RS4BB
6921 036476 110237 036721          MOV     R2,RS4HH
6922 036502 110237 036765          MOV     R2,RS4NN
6923
6924 036506 013703 037156          MOV     DRS4T1,R3      ;SET UP THE DISK ADDRESS.
6925 036512 013704 037160          MOV     DRS4T2,R4
6926 036516 010337 036656          MOV     R3,RS4CC
6927 036522 010337 037030          MOV     R3,RS4113
6928 036526 010337 036722          MOV     R3,RS411
6929 036532 010337 036766          MOV     R3,RS400
6930 036536 010437 036660          MOV     R4,RS4DD
6931 036542 010437 036724          MOV     R4,RS4JJ
6932 036546 010437 037032          MOV     R4,RS4114
6933 036552 010437 036770          MOV     R4,RS4PP
6934
6935 036556 010137 036634          MOV     R1,RS4AA1      ;SET THE MEMORY ADDRESS.
6936 036562 010137 036662          MOV     R1,RS4EE
6937 036566 010137 036726          MOV     R1,RS4KK
6938 036572 010137 036772          MOV     R1,RS4QQ
6939 036576 010137 037034          MOV     R1,RS4115
6940 036602 010037 036774          MOV     R0,RS4RR
6941 036606 010037 037036          MOV     R0,RS4116
6942 036612 010037 036636          MOV     R0,RS4AA2
6943 036616 010037 036664          MOV     R0,RS4FF
6944 036622 010037 036730          MOV     R0,RS4LL
6945
6946 036626 104413          RESREG
6947
6948 036630 104425          WRRAND      ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
6949 036632 000000  RS4AA3: .WORD 0
6950 036634 000000  RS4AA1: .WORD 0
6951 036636 000000  RS4AA2: .WORD 0
6952 036640 004000          .WORD 4000
6953 036642 005237 036632  INC     RS4AA3
6954
6955 036646 000240          NOP
6956 036650 000237          SPL     7
    
```

```

6957 036652 104426          CALRS4          ;GET THE RS4 TO DO THE TRANSFER FROM MEMORY
6958 036654      161          .BYTE          161
6959 036655      000          RS4BB: .BYTE          0
6960 036656 000000          RS4CC: .WORD          0
6961 036660 000000          RS4DD: .WORD          0
6962 036662 000000          RS4EE: .WORD          0
6963 036664 000000          RS4FF: .WORD          0
6964 036666 004000          .WORD          4000
6965 036670 036706          .WORD          RS4GG
6966
6967 036672 000240          NOP
6968 036674 004737 037104          JSR          PC,RS4YY
6969 036700 005066 000002          CLR          2(SP)
6970 036704 000002          RTI          ;GO DO SOMETHING ELSE WHILE WAITING
6971                                     ;FOR THE INTERRUPT!
6972
6973 036706 000240          RS4GG: NOP
6974 036710 004737 037104          JSR          PC,RS4YY          ;SEE IF THERE WERE ANY ERRORS.
6975
6976 036714 000237          SPL          7
6977 036716 104426          CALRS4          ;DO THE WRITE CHECK
6978 036720      151          .BYTE          151
6979 036721      000          RS4HH: .BYTE          0
6980 036722 000000          RS4II: .WORD          0
6981 036724 000000          RS4JJ: .WORD          0
6982 036726 000000          RS4KK: .WORD          0
6983 036730 000000          RS4LL: .WORD          0
6984 036732 004000          .WORD          4000
6985 036734 036752          .WORD          RS4MM
6986
6987 036736 000240          NOP
6988 036740 004737 037104          JSR          PC,RS4YY
6989 036744 005066 000002          CLR          2(SP)
6990 036750 000002          RTI          ;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
6991
6992 036752 000240          RS4MM: NOP
6993 036754 004737 037104          JSR          PC,RS4YY          ;SEE IF THERE WERE ANY ERRORS.
6994
6995
6996 036760 000237          SPL          7
6997 036762 104426          CALRS4          ;READ THE DISK.
6998 036764      171          .BYTE          171
6999 036765      000          RS4NN: .BYTE          0
7000 036766 000000          RS4OO: .WORD          0
7001 036770 000000          RS4PP: .WORD          0
7002 036772 000000          RS4QQ: .WORD          0
7003 036774 000000          RS4RR: .WORD          0
7004 036776 004000          .WORD          4000
7005 037000 037016          .WORD          RS4111
7006
7007 037002 000240          NOP
7008 037004 004737 037104          JSR          PC,RS4YY
7009 037010 005066 000002          CLR          2(SP)
7010 037014 000002          RTI          ;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
7011
7012 037016 004737 037104          RS4111: JSR          PC,RS4YY
    
```

```

7013 037022 000237          SPL      7
7014
7015 037024 104426          CALRS4
7016 037026      151          .BYTE   151
7017 037027      000          RS4112: .BYTE   0
7018 037030 000000          RS4113: .WORD   0
7019 037032 000000          RS4114: .WORD   0
7020 037034 000000          RS4115: .WORD   0
7021 037036 000000          RS4116: .WORD   0
7022 037040 004000          .WORD  4000
7023 037042 037060          .WORD  RS4SS
7024 037044 000240          NOP
7025 037046 004737 037104    JSR     PC,RS4YY
7026 037052 005066 000002    CLR     2(SP)
7027 037056 000002          RTI
7028
7029 037060 000240          RS4SS:  NOP
7030 037062 004737 037104    JSR     PC,RS4YY      ;SEE IF ANY ERRORS OCCURRED.
7031
7032 037066 005337 036234    DEC     RS4CR          ;DECURMENT THE PASS COUNT.
7033 037072 001001          BNE     RS4XX          ;IF NOT DONE CONTINUE.
7034 037074 000002          RTI                    ;IF DONE GET OUT!
7035
7036 037076 000240          RS4XX:  NOP
7037 037100 000137 036374    JMP     RS4AA          ;RESTART.
7038
7039 037104 000240          RS4YY:  NOP
7040 037106 005737 061004    TST     RS4ER1        ;SEE IF ANY ERRORS OCCURRED.
7041 037112 001420          BEQ     RS4ZZ          ;IF NOT THEN RETURN TO CALL.
7042
7043 037114 000237          SPL      7
7044 037116 005037 036234    CLR     RS4CR          ;IF YES THEN CLEAR THE PASS COUNT.
7045 037122 013737 061006 001634  MOV     RS4ER2,$TMP1  ;AND MAKE AN ERROR CALL.
7046 037130 013737 061012 001640  MOV     RS4ER4,$TMP3
7047 037136 013737 061010 001636  MOV     RS4ER3,$TMP2
7048 037144 104154          ERROR   154
7049 037146 000230          SPL      0
7050 037150 005726          TST     (SP)+
7051 037152 000002          RTI                    ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
7052
7053
7054 037154 000207          RS4ZZ:  RTS     PC      ;THERE WERE NO ERRORS.
7055
7056 037156 000000          DRS4T1: .WORD   0
7057 037160 000000          DRS4T2: .WORD   0
7058
7059
7060
7061
7062
7063
7064 037162 000240          ;THIS IS THE RP4 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
7065 037164 012737 004004 037770  DRRP4:  NOP          ;TEST.
7066 037172 012737 003003 037772  MOV     #4004,DRP4T1  ;INITIALIZE THE RANDOM DISK ADDRESS
7067 037200 012737 002002 037444  MOV     #3003,DRP4T2  ;GENERATER.
7068

```

```

7069 037206 000240          RP4AA:  NOP
7070 037210 000240          NOP
7071 037212 104412          SAVREG
7072 037214 004737 042232  JSR    PC,GETBUF      ;GET A MEMORY BUFFER.
7073 037220 036274          .WORD  RP4RB
7074 037222 013701 036274  MOV    RP4RB,R1
7075 037226 005000          CLR    R0
7076 037230 073027 000014  ASHC   #12.,R0
7077
7078 037234 000237          SPL    7              ;GET A RANDOM DISK ADDRESS.
7079 037236 013737 037770 054370  MOV    DRP4T1,$HINUM
7080 037244 013737 037772 054372  MOV    DRP4T2,$LONUM
7081 037252 004737 054272          JSR    PC,$RAND
7082 037256 013737 054370 037770  MOV    $HINUM,DRP4T1
7083 037264 013737 054372 037772  MOV    $LONUM,DRP4T2
7084 037272 000230          SPL    0
7085
7086 037274 013702 036250          MOV    RP4SUN,R2      ;SET UP THE DEVICE UNIT NUM.
7087 037300 110237 037641          MOVB   R2,RP4I12
7088 037304 110237 037467          MOVB   R2,RP4BB
7089 037310 110237 037533          MOVB   R2,RP4HH
7090 037314 110237 037577          MOVB   R2,RP4NN
7091
7092 037320 013703 037770          MOV    DRP4T1,R3      ;SET UP THE DISK ADDRESS.
7093 037324 013704 037772          MOV    DRP4T2,R4
7094 037330 010337 037470          MOV    R3,RP4CC
7095 037334 010337 037642          MOV    R3,RP4I13
7096 037340 010337 037534          MOV    R3,RP4I1
7097 037344 010337 037600          MOV    R3,RP4O0
7098 037350 010437 037472          MOV    R4,RP4DD
7099 037354 010437 037536          MOV    R4,RP4JJ
7100 037360 010437 037644          MOV    R4,RP4I14
7101 037364 010437 037602          MOV    R4,RP4PP
7102
7103 037370 010137 037446          MOV    R1,RP4AA1     ;SET THE MEMORY ADDRESS.
7104 037374 010137 037474          MOV    R1,RP4EE
7105 037400 010137 037540          MOV    R1,RP4KK
7106 037404 010137 037604          MOV    R1,RP4QQ
7107 037410 010137 037646          MOV    R1,RP4I15
7108 037414 010037 037606          MOV    R0,RP4RR
7109 037420 010037 037650          MOV    R0,RP4I16
7110 037424 010037 037450          MOV    R0,RP4AA2
7111 037430 010037 037476          MOV    R0,RP4FF
7112 037434 010037 037542          MOV    R0,RP4LL
7113
7114 037440 104413          RESREG
7115
7116 037442 104425          WRRAND                ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
7117 037444 000000  RP4AA3: .WORD 0
7118 037446 000000  RP4AA1: .WORD 0
7119 037450 000000  RP4AA2: .WORD 0
7120 037452 004000          .WORD 4000
7121 037454 005237 037444  INC    RP4AA3
7122
7123 037460 000240          NOP
7124 037462 000237          SPL    7
    
```

```

7125 037464 104427          CALRP4          ;GET THE RP4 TO DO THE TRANSFER FROM MEMORY
7126 037466      161          .BYTE          161
7127 037467      000          RP4BB: .BYTE          0
7128 037470 000000          RP4CC: .WORD          0
7129 037472 000000          RP4DD: .WORD          0
7130 037474 000000          RP4EE: .WORD          0
7131 037476 000000          RP4FF: .WORD          0
7132 037500 004000          .WORD          4000
7133 037502 037520          .WORD          RP4GG
7134
7135 037504 000240          NOP
7136 037506 004737 037716  JSR          PC,RP4YY
7137 037512 005066 000002  CLR          2(SP)
7138 037516 000002          RTI          ;GO DO SOMETHING ELSE WHILE WAITING
7139                                     ;FOR THE INTERRUPT!
7140
7141 037520 000240          RP4GG: NOP
7142 037522 004737 037716  JSR          PC,RP4YY          ;SEE IF THERE WERE ANY ERRORS.
7143
7144 037526 000237          SPL          7
7145 037530 104427          CALRP4          ;DO THE WRITE CHECK
7146 037532      151          .BYTE          151
7147 037533      000          RP4HH: .BYTE          0
7148 037534 000000          RP4II: .WORD          0
7149 037536 000000          RP4JJ: .WORD          0
7150 037540 000000          RP4KK: .WORD          0
7151 037542 000000          RP4LL: .WORD          0
7152 037544 004000          .WORD          4000
7153 037546 037564          .WORD          RP4MM
7154
7155 037550 000240          NOP
7156 037552 004737 037716  JSR          PC,RP4YY
7157 037556 005066 000002  CLR          2(SP)
7158 037562 000002          RTI          ;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
7159
7160 037564 000240          RP4MM: NOP
7161 037566 004737 037716  JSR          PC,RP4YY          ;SEE IF THERE WERE ANY ERRORS.
7162
7163
7164 037572 000237          SPL          7
7165 037574 104427          CALRP4          ;READ THE DISK.
7166 037576      171          .BYTE          171
7167 037577      000          RP4NN: .BYTE          0
7168 037600 000000          RP4OO: .WORD          0
7169 037602 000000          RP4PP: .WORD          0
7170 037604 000000          RP4QQ: .WORD          0
7171 037606 000000          RP4RR: .WORD          0
7172 037610 004000          .WORD          4000
7173 037612 037630          .WORD          RP4111
7174
7175 037614 000240          NOP
7176 037616 004737 037716  JSR          PC,RP4YY
7177 037622 005066 000002  CLR          2(SP)
7178 037626 000002          RTI          ;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
7179
7180 037630 004737 037716  RP4111: JSR          PC,RP4YY
    
```

```

7181 037634 000237          SPL      7
7182
7183 037636 104427          CALRP4
7184 037640      151          .BYTE   151
7185 037641      000          RP4112: .BYTE   0
7186 037642 000000          RP4113: .WORD   0
7187 037644 000000          RP4114: .WORD   0
7188 037646 000000          RP4115: .WORD   0
7189 037650 000000          RP4116: .WORD   0
7190 037652 004000          .WORD  4000
7191 037654 037672          .WORD  RP4SS
7192 037656 000240          NOP
7193 037660 004737 037716          JSR    PC,RP4YY
7194 037664 005066 000002          CLR    2(SP)
7195 037670 000002          RTI
7196
7197 037672 000240          RP4SS: NOP
7198 037674 004737 037716          JSR    PC,RP4YY      ;SEE IF ANY ERRORS OCCURRED.
7199
7200 037700 005337 036236          DEC    RP4CR          ;DECRIMENT THE PASS COUNT.
7201 037704 001001          BNE    RP4XX          ;IF NOT DONE CONTINUE.
7202 037706 000002          RTI          ;IF DONE GET OUT!
7203
7204 037710 000240          RP4XX: NOP
7205 037712 000137 037206          JMP    RP4AA          ;RESTART.
7206
7207 037716 000240          RP4YY: NOP
7208 037720 005737 060034          TST    RP4ER1      ;SEE IF ANY ERRORS OCCURRED.
7209 037724 001420          BEQ    RP4ZZ          ;IF NOT THEN RETURN TO CALL.
7210
7211 037726 000237          SPL      7
7212 037730 005037 036236          CLR    RP4CR          ;IF YES THEN CLEAR THE PASS COUNT.
7213 037734 013737 060036 001634          MOV    RP4ER2,$TMP1 ;AND MAKE AN ERROR CALL.
7214 037742 013737 060042 001640          MOV    RP4ER4,$TMP3
7215 037750 013737 060040 001636          MOV    RP4ER3,$TMP2
7216 037756 104155          ERROR  155
7217 037760 000230          SPL      0
7218 037762 005726          TST    (SP)+
7219 037764 000002          RTI          ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
7220
7221
7222 037766 000207          RP4ZZ: RTS    PC          ;THERE WERE NO ERRORS.
7223
7224 037770 000000          DRP4T1: .WORD  0
7225 037772 000000          DRP4T2: .WORD  0
7226
7227
7228
7229          ;THIS IS THE RH4 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
7230          ;TEST.
7231
7232 037774 000240          DRRH4: NOP
7233 037776 012737 070070 040562          MOV    #70070,DRH4T1 ;INITIALIZE THE RANDOM DISK ADDRESS
7234 040004 012737 060060 040564          MOV    #60060,DRH4T2 ;GENERATER.
7235 040012 012737 050050 040236          MOV    #50050,RH4AA3
7236

```

```

7237 040020 000240          RH4AA: NOP
7238 040022 000240          NOP
7239 040024 104412          SAVREG
7240 040026 004737 042232   JSR    PC,GETBUF      ;GET A MEMORY BUFFER.
7241 040032 036276          .WORD  RH4RB
7242 040034 013701 036276   MOV    RH4RB,R1
7243 040040 005000          CLR    R0
7244 040042 073027 000014   ASHC  #12.,R0
7245
7246 040046 000237          SPL    7              ;GET A RANDOM DISK ADDRESS.
7247 040050 013737 040562 054370  MOV    DRH4T1,$HINUM
7248 040056 013737 040564 054372  MOV    DRH4T2,$LONUM
7249 040064 004737 054272   JSR    PC,$RAND
7250 040070 013737 054370 040562  MOV    $HINUM,DRH4T1
7251 040076 013737 054372 040564  MOV    $LONUM,DRH4T2
7252 040104 000230          SPL    0
7253
7254 040106 013702 036252   MOV    RH4SUN,R2      ;SET UP THE DEVICE UNIT NUM.
7255 040112 110237 040433   MOVB  R2,RH4112
7256 040116 110237 040261   MOVB  R2,RH4BB
7257 040122 110237 040325   MOVB  R2,RH4HH
7258 040126 110237 040371   MOVB  R2,RH4NN
7259
7260 040132 013703 040562   MOV    DRH4T1,R3      ;SET UP THE DISK ADDRESS.
7261 040136 013704 040564   MOV    DRH4T2,R4
7262 040142 010337 040262   MOV    R3,RH4CC
7263 040146 010337 040434   MOV    R3,RH4113
7264 040152 010337 040326   MOV    R3,RH4II
7265 040156 010337 040372   MOV    R3,RH4OO
7266
7267 040162 010137 040240   MOV    R1,RH4AA1      ;SET THE MEMORY ADDRESS.
7268 040166 010137 040266   MOV    R1,RH4EE
7269 040172 010137 040332   MOV    R1,RH4KK
7270 040176 010137 040376   MOV    R1,RH4QQ
7271 040202 010137 040440   MOV    R1,RH4115
7272 040206 010037 040400   MOV    R0,RH4RR
7273 040212 010037 040442   MOV    R0,RH4116
7274 040216 010037 040242   MOV    R0,RH4AA2
7275 040222 010037 040270   MOV    R0,RH4FF
7276 040226 010037 040334   MOV    R0,RH4LL
7277
7278 040232 104413          RESREG
7279
7280 040234 104425          WRRAND      ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
7281 040236 000000          RH4AA3: .WORD  0
7282 040240 000000          RH4AA1: .WORD  0
7283 040242 000000          RH4AA2: .WORD  0
7284 040244 004000          .WORD  4000
7285 040246 005237 040236   INC    RH4AA3
7286
7287 040252 000240          NOP
7288 040254 000237          SPL    7
7289 040256 104430          CALRH4      ;GET THE RH4 TO DO THE TRANSFER FROM MEMORY
7290 040260 161          .BYTE  161
7291 040261 000          RH4BB: .BYTE  0
7292 040262 000000          RH4CC: .WORD  0
    
```



```

7293 040264 000000      RH4DD: .WORD 0
7294 040266 000000      RH4EE: .WORD 0
7295 040270 000000      RH4FF: .WORD 0
7296 040272 004000      .WORD 4000
7297 040274 040312      .WORD RH4GG
7298
7299 040276 000240      NOP
7300 040300 004737 040510      JSR PC,RH4YY
7301 040304 005066 000002      CLR 2(SP)
7302 040310 000002      RTI      ;GO DO SOMETHING ELSE WHILE WAITING
7303                                     ;FOR THE INTERRUPT!
7304
7305 040312 000240      RH4GG: NOP
7306 040314 004737 040510      JSR PC,RH4YY      ;SEE IF THERE WERE ANY ERRORS.
7307
7308 040320 000237      SPL 7
7309 040322 104430      CALRH4      ;DO THE WRITE CHECK
7310 040324 171      .BYTE 171
7311 040325 000      RH4HH: .BYTE 0
7312 040326 000000      RH4II: .WORD 0
7313 040330 000000      RH4JJ: .WORD 0
7314 040332 000000      RH4KK: .WORD 0
7315 040334 000000      RH4LL: .WORD 0
7316 040336 004000      .WORD 4000
7317 040340 040356      .WORD RH4MM
7318
7319 040342 000240      NOP
7320 040344 004737 040510      JSR PC,RH4YY
7321 040350 005066 000002      CLR 2(SP)
7322 040354 000002      RTI      ;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
7323
7324 040356 000240      RH4MM: NOP
7325 040360 004737 040510      JSR PC,RH4YY      ;SEE IF THERE WERE ANY ERRORS.
7326
7327
7328 040364 000237      SPL 7
7329 040366 104430      CALRH4      ;READ THE DISK.
7330 040370 151      .BYTE 151
7331 040371 000      RH4NN: .BYTE 0
7332 040372 000000      RH4OO: .WORD 0
7333 040374 000000      RH4PP: .WORD 0
7334 040376 000000      RH4QQ: .WORD 0
7335 040400 000000      RH4RR: .WORD 0
7336 040402 004000      .WORD 4000
7337 040404 040422      .WORD RH4111
7338
7339 040406 000240      NOP
7340 040410 004737 040510      JSR PC,RH4YY
7341 040414 005066 000002      CLR 2(SP)
7342 040420 000002      RTI      ;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
7343
7344 040422 004737 040510      RH4111: JSR PC,RH4YY
7345 040426 000237      SPL 7
7346
7347 040430 104430      CALRH4
7348 040432 171      .BYTE 171

```

```

7349 040433 000          RH4112: .BYTE 0
7350 040434 000000      RH4113: .WORD 0
7351 040436 000000      RH4114: .WORD 0
7352 040440 000000      RH4115: .WORD 0
7353 040442 000000      RH4116: .WORD 0
7354 040444 004000      .WORD 4000
7355 040446 040464      .WORD RH45S
7356 040450 000240      NOP
7357 040452 004737 040510 JSR PC,RH4YY
7358 040456 005066 000002 CLR 2(SP)
7359 040462 000002      RTI
7360
7361 040464 000240      RH45S: NOP
7362 040466 004737 040510 JSR PC,RH4YY ;SEE IF ANY ERRORS OCCURRED.
7363
7364 040472 005337 036240 DEC RH4CR ;DECRIEMENT THE PASS COUNT.
7365 040476 001001      BNE RH4XX ;IF NOT DONE CONTINUE.
7366 040500 000002      RTI ;IF DONE GET OUT!
7367
7368 040502 000240      RH4XX: NOP
7369 040504 000137 040020 JMP RH4AA ;RESTART.
7370
7371 040510 000240      RH4YY: NOP
7372 040512 005737 063500 TST RH4ER1 ;SEE IF ANY ERRORS OCCURRED.
7373 040516 001420      BEQ RH4ZZ ;IF NOT THEN RETURN TO CALL.
7374
7375 040520 000237      SPL 7
7376 040522 005037 036240 CLR RH4CR ;IF YES THEN CLEAR THE PASS COUNT.
7377 040526 013737 063502 001634 MOV RH4ER2,$TMP1 ;AND MAKE AN ERROR CALL.
7378 040534 013737 063506 001640 MOV RH4ER4,$TMP3
7379 040542 013737 063504 001636 MOV RH4ER3,$TMP2
7380 040550 104156      ERROR 156
7381 040552 000230      SPL 0
7382 040554 005726      TST (SP)+
7383 040556 000002      RTI ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
7384 ;FROM THE TEST.
7385
7386 040560 000207      RH4ZZ: RTS PC ;THERE WERE NO ERRORS.
7387
7388 040562 000000      DRH4T1: .WORD 0
7389 040564 000000      DRH4T2: .WORD 0
7390
7391
7392
7393
7394 ;THIS IS THE RK5 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
7395 ;TEST.
7396
7397 040566 000240      DRRK5: NOP
7398 040570 012737 030030 041374 MOV #30030,DRK5T1 ;INITIALIZE THE RANDOM DISK ADDRESS
7399 040576 012737 040040 041376 MOV #40040,DRK5T2 ;GENERATER.
7400 040604 012737 050050 041050 MOV #50050,RK5AA3
7401
7402 040612 000240      RK5AA: NOP
7403 040614 000240      NOP
7404 040616 104412      SAVREG

```

```

7405 040620 004737 042232      JSR    PC,GETBUF      ;GET A MEMORY BUFFER.
7406 040624 036300              .WORD  RK5RB
7407 040626 013701 036300      MOV    RK5RB,R1
7408 040632 005000              CLR    R0
7409 040634 073027 000014      ASHC   #12.,R0
7410
7411 040640 000237              SPL    7              ;GET A RANDOM DISK ADDRESS.
7412 040642 013737 041374 054370  MOV    DRK5T1,$HINUM
7413 040650 013737 041376 054372  MOV    DRK5T2,$LONUM
7414 040656 004737 054272      JSR    PC,$RAND
7415 040662 013737 054370 041374  MOV    $HINUM,DRK5T1
7416 040670 013737 054372 041376  MOV    $LONUM,DRK5T2
7417 040676 000230              SPL    0
7418
7419 040700 013702 036254      MOV    RK5SUN,R2      ;SET UP THE DEVICE UNIT NUM.
7420 040704 110237 041245      MOVB   R2,RK5I12
7421 040710 110237 041073      MOVB   R2,RK5BB
7422 040714 110237 041137      MOVB   R2,RK5HH
7423 040720 110237 041203      MOVB   R2,RK5NN
7424
7425 040724 013703 041374      MOV    DRK5T1,R3      ;SET UP THE DISK ADDRESS.
7426 040730 013704 041376      MOV    DRK5T2,R4
7427 040734 010337 041074      MOV    R3,RK5CC
7428 040740 010337 041246      MOV    R3,RK5I13
7429 040744 010337 041140      MOV    R3,RK5II
7430 040750 010337 041204      MOV    R3,RK5OO
7431 040754 010437 041076      MOV    R4,RK5DD
7432 040760 010437 041142      MOV    R4,RK5JJ
7433 040764 010437 041250      MOV    R4,RK5I14
7434 040770 010437 041206      MOV    R4,RK5PP
7435
7436 040774 010137 041052      MOV    R1,RK5AA1     ;SET THE MEMORY ADDRESS.
7437 041000 010137 041100      MOV    R1,RK5EE
7438 041004 010137 041144      MOV    R1,RK5KK
7439 041010 010137 041210      MOV    R1,RK5QQ
7440 041014 010137 041252      MOV    R1,RK5I15
7441 041020 010037 041212      MOV    R0,RK5RR
7442 041024 010037 041254      MOV    R0,RK5I16
7443 041030 010037 041054      MOV    R0,RK5AA2
7444 041034 010037 041102      MOV    R0,RK5FF
7445 041040 010037 041146      MOV    R0,RK5LL
7446
7447 041044 104413              RESREG
7448
7449 041046 104425              WRRAND              ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
7450 041050 000000      RK5AA3: .WORD  0
7451 041052 000000      RK5AA1: .WORD  0
7452 041054 000000      RK5AA2: .WORD  0
7453 041056 004000              .WORD  4000
7454 041060 005237 041050      INC    RK5AA3
7455
7456 041064 000240              NOP
7457 041066 000237              SPL    7
7458 041070 104431              CALRK5              ;GET THE RK5 TO DO THE TRANSFER FROM MEMORY
7459 041072 103              .BYTE  103
7460 041073 000      RK5BB: .BYTE  0

```

```

7461 041074 000000          RK5CC: .WORD 0
7462 041076 000000          RK5DD: .WORD 0
7463 041100 000000          RK5EE: .WORD 0
7464 041102 000000          RK5FF: .WORD 0
7465 041104 004000          .WORD 4000
7466 041106 041124          .WORD RK5GG
7467
7468 041110 000240          NOP
7469 041112 004737 041322  JSR    PC,RK5YY
7470 041116 005066 000002  CLR    2(SP)
7471 041122 000002          RTI
                                ;GO DO SOMETHING ELSE WHILE WAITING
                                ;FOR THE INTERRUPT!
7472
7473
7474 041124 000240          RK5GG: NOP
7475 041126 004737 041322  JSR    PC,RK5YY          ;SEE IF THERE WERE ANY ERRORS.
7476
7477 041132 000237          SPL    7
7478 041134 104431          CALRK5          ;DO THE WRITE CHECK
7479 041136 107          .BYTE 107
7480 041137 000          RK5HH: .BYTE 0
7481 041140 000000          RK5II: .WORD 0
7482 041142 000000          RK5JJ: .WORD 0
7483 041144 000000          RK5KK: .WORD 0
7484 041146 000000          RK5LL: .WORD 0
7485 041150 004000          .WORD 4000
7486 041152 041170          .WORD RK5MM
7487
7488 041154 000240          NOP
7489 041156 004737 041322  JSR    PC,RK5YY
7490 041162 005066 000002  CLR    2(SP)
7491 041166 000002          RTI
                                ;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
7492
7493 041170 000240          RK5MM: NOP
7494 041172 004737 041322  JSR    PC,RK5YY          ;SEE IF THERE WERE ANY ERRORS.
7495
7496
7497 041176 000237          SPL    7
7498 041200 104431          CALRK5          ;READ THE DISK.
7499 041202 105          .BYTE 105
7500 041203 000          RK5NN: .BYTE 0
7501 041204 000000          RK5OO: .WORD 0
7502 041206 000000          RK5PP: .WORD 0
7503 041210 000000          RK5QQ: .WORD 0
7504 041212 000000          RK5RR: .WORD 0
7505 041214 004000          .WORD 4000
7506 041216 041234          .WORD RK5111
7507
7508 041220 000240          NOP
7509 041222 004737 041322  JSR    PC,RK5YY
7510 041226 005066 000002  CLR    2(SP)
7511 041232 000002          RTI
                                ;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
7512
7513 041234 004737 041322  RK5111: JSR    PC,RK5YY
7514 041240 000237          SPL    7
7515
7516 041242 104431          CALRK5

```

```

7517 041244 107 .BYTE 107
7518 041245 000 RK5112: .BYTE 0
7519 041246 000000 RK5113: .WORD 0
7520 041250 000000 RK5114: .WORD 0
7521 041252 000000 RK5115: .WORD 0
7522 041254 000000 RK5116: .WORD 0
7523 041256 004000 .WORD 4000
7524 041260 041276 .WORD RK5SS
7525 041262 000240 NOP
7526 041264 004737 041322 JSR PC,RK5YY
7527 041270 005066 000002 CLR 2(SP)
7528 041274 000002 RTI
7529
7530 041276 000240 RK5SS: NOP
7531 041300 004737 041322 JSR PC,RK5YY ;SEE IF ANY ERRORS OCCURRED.
7532
7533 041304 005337 036242 DEC RK5CR ;DECRIMENT THE PASS COUNT.
7534 041310 001001 BNE RK5XX ;IF NOT DONE CONTINUE.
7535 041312 000002 RTI ;IF DONE GET OUT!
7536
7537 041314 000240 RK5XX: NOP
7538 041316 000137 040612 JMP RK5AA ;RESTART.
7539
7540 041322 000240 RK5YY: NOP
7541 041324 005737 061740 TST RK5ER1 ;SEE IF ANY ERRORS OCCURRED.
7542 041330 001420 BEQ RK5ZZ ;IF NOT THEN RETURN TO CALL.
7543
7544 041332 000237 SPL 7
7545 041334 005037 036242 CLR RK5CR ;IF YES THEN CLEAR THE PASS COUNT.
7546 041340 013737 061742 001634 MOV RK5ER2,$TMP1 ;AND MAKE AN ERROR CALL.
7547 041346 013737 061746 001640 MOV RK5ER4,$TMP3
7548 041354 013737 061744 001636 MOV RK5ER3,$TMP2
7549 041362 104160 ERROR 160
7550 041364 000230 SPL 0
7551 041366 005726 TST (SP)+
7552 041370 000002 RTI ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
7553 ;FROM THE TEST.
7554
7555 041372 000207 RK5ZZ: RTS PC ;THERE WERE NO ERRORS.
7556
7557 041374 000000 DRK5T1: .WORD 0
7558 041376 000000 DRK5T2: .WORD 0
7559
7560
7561
7562 ;THIS IS THE UBE DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
7563 ;TEST.
7564 041400 012737 050050 041746 DRUBE: MOV #50050,DUBET1 ;INITIALIZE THE RANDOM DATA
7565 041406 012737 060060 041750 MOV #60060,DUBET2 ;GENERATER.
7566 041414 012737 070070 041554 MOV #70070,UBEAA3
7567
7568 041422 104412 UBEAA: SAVREG
7569 041424 004737 042232 JSR PC,GETBUF ;PICK UP A MEMORY BUFFER
7570 041430 036302 .WORD UBERB
7571
7572 041432 013701 036302 MOV UBERB,R1 ;COMPUTE THE MEMORY ADDRESS.
    
```

```

7573 041436 005000          CLR      R0
7574 041440 073027 000014  ASHC    #12.,R0
7575 041444 010137 041556  MOV     R1,UBEAA1
7576 041450 010137 041602  MOV     R1,UBEDD
7577 041454 010137 041642  MOV     R1,UBEII
7578 041460 010037 041560  MOV     R0,UBEAA2
7579 041464 010037 041604  MOV     R0,UBEED
7580 041470 010037 041644  MOV     R0,UBEJJ
7581
7582 041474 000237          SPL      7
7583 041476 013737 041746 054370  MOV     DUBET1,$HINUM
7584 041504 013737 041750 054372  MOV     DUBET2,$LONUM
7585 041512 004737 054272          JSR     PC,$RAND
7586 041516 013737 054370 041746  MOV     $HINUM,DUBET1
7587 041524 013737 054372 041750  MOV     $LONUM,DUBET2
7588 041532 000230          SPL      0
7589
7590 041534 013703 041746          MOV     DUBET1,R3          ;SET THE UNIBUS TESTER DATA REG.
7591 041540 010337 041640          MOV     R3,UBEHH
7592 041544 010337 041600          MOV     R3,UBECCC
7593
7594 041550 104413          RESREG
7595
7596 041552 104425          WRRAND          ;FILL THE MEMORY BUFFER WITH
7597 041554 000000  UBEEA3: .WORD 0          ;RANDOM DATA.
7598 041556 000000  UBEEA1: .WORD 0
7599 041560 000000  UBEEA2: .WORD 0
7600 041562 004000          .WORD 4000
7601 041564 005237 041554  INC     UBEEA3
7602
7603 041570 000237          SPL      7
7604 041572 104432          CALUBE          ;DO A READ MEMORY FUNCTION.
7605 041574 042543          .WORD 42543
7606 041576 000000  UBEBB: .WORD 0
7607 041600 000000  UBEECC: .WORD 0
7608 041602 000000  UBEDD: .WORD 0
7609 041604 000000  UBEEE: .WORD 0
7610 041606 010000          .WORD 10000
7611
7612 041610 041624          .WORD UBEEFF
7613
7614 041612 004737 041704          JSR     PC,UBEYY
7615 041616 005066 000002          CLR     2(SP)
7616 041622 000002          RTI
7617
7618 041624 004737 041704  UBEEFF: JSR     PC,UBEYY          ;GO DO SOMETHING ELSE WHILE
7619
7620 041630 000237          SPL      7          ;DO A WRITE MEMORY FUNCTION.
7621 041632 104432          CALUBE
7622 041634 042543          .WORD 42543
7623 041636 000000  UBEGG: .WORD 0
7624 041640 000000  UBEHH: .WORD 0
7625 041642 000000  UBEII: .WORD 0
7626 041644 000000  UBEJJ: .WORD 0
7627 041646 010000          .WORD 10000
7628 041650 041664          .WORD UBEKK
    
```

```

7629
7630 041652 004737 041704      JSR    PC,UBEYY
7631 041656 005066 000002      CLR    2(SP)
7632 041662 000002                RTI    ;GO DO SOMETHING ELSE WHILE
7633                                ;WAITING FOR THE INTERRUPT.
7634 041664 004737 041704      UBEKK: JSR    PC,UBEYY
7635
7636 041670 005337 036244      DEC    UBECR ;DECREMENT THE PASS COUNT.
7637 041674 001001                BNE    UBELL ;BR IF NOT DONE
7638
7639 041676 000002                RTI    ;IF DONE RETURN.
7640 041700 000137 041422      UBELL: JMP    UBEEA ;IF NOT DONE DO ANOTHER PASS.
7641
7642 041704 005737 062754      UBEYY: TST    UBEER1 ;WERE THERE ANY ERRORS?
7643 041710 001415                BEQ    UBEZZ ;BR IF NO.
7644
7645 041712 000237                SPL    7 ;IF THERE WERE REPORT DEVICE FAILURE.
7646 041714 005037 036244      CLR    UBECR
7647 041720 013737 062756 001634  MOV    UBEER2,$TMP1
7648 041726 013737 062760 001636  MOV    UBEER3,$TMP2
7649 041734 104161                ERROR  161
7650 041736 005726                TST    (SP)+
7651 041740 000230                SPL    0
7652 041742 000002                RTI    ;RETURN WITH THIS DRIVER LOCKED OUT.
7653 041744 000207      UBEZZ: RTS    PC ;NO ERRORS CONTINUE.
7654
7655 041746 000000      DUBET1: .WORD 0
7656 041750 000000      DUBET2: .WORD 0
7657
7658
7659
7660
7661      ;THIS ROUTINE IS USED TO GENERATE A BUFFER FULL OF RANDOM DATA.
7662      ;IT IS CALLED USING THE TRAP TABLE CALL:
7663      ;
7664      ;
7665      ;
7666      ;
7667      ;
7668      ;
7669      ;
7670      ;
7671      ;
7672      ;
7673 041752 000237      RANDWR: SPL    7
7674 041754 011637 042114      MOV    (SP),RANDTP
7675 041760 062716 000010      ADD    #10,(SP)
7676 041764 104412                SAVREG
7677 041766 013700 042114      MOV    RANDTP,R0
7678 041772 012001                MOV    (R0)+,R1
7679 041774 012002                MOV    (R0)+,R2
7680 041776 012003                MOV    (R0)+,R3
7681 042000 012004                MOV    (R0)+,R4
7682 042002 010237 042112      MOV    R2,RLWT
7683 042006 010337 042110      MOV    R3,RHWT
7684 042012 010137 054370      MOV    R1,$HINUM
    
```

```

7685 042016 005037 054372          CLR      $LONUM
7686
7687 042022 013702 042110          1$:     MOV      RHWT,R2          ;COMPUTE THE VIRTUAL ADDRESS OF THE BUFFER WORD.
7688 042026 013703 042112          MOV      RLWT,R3
7689 042032 073227 177772          ASHC     #-6,R2
7690 042036 010337 172354          MOV      R3,@WKIPAR6
7691 042042 013702 042112          MOV      RLWT,R2
7692 042046 042702 177700          BIC      #177700,R2
7693 042052 062702 140000          ADD      #140000,R2
7694 042056 004737 054272          JSR      PC,$RAND
7695 042062 013712 054370          MOV      $HINUM,(R2)
7696 042066 062737 000002 042112  ADD      #2,RLWT
7697 042074 005537 042110          ADC      RHWT
7698 042100 077430          SOB      R4,1$
7699
7700 042102 000230          SPL      0
7701 042104 104413          RESREG
7702 042106 000002          RTI
7703
7704 042110 000000          RHWT:    .WORD  0
7705 042112 000000          RLWT:    .WORD  0
7706 042114 000000          RANDTP:  .WORD  0
7707
7708          ;THIS ROUTINE IS USED TO INITIALIZE THE GET BUFFER ROUTINE.
7709 042116 012700 036272  GTBINT:  MOV      #RS4RB,R0          ;CLEAR ALL THE BUFFER POINTERS.
7710 042122 012701 000005          MOV      #5,R1
7711
7712 042126 005020          1$:     CLR      (R0)+
7713 042130 077102          SOB      R1,1$
7714 042132 104417          SIZE
7715 042134 000000          ;COMPUTE THE SIZE OF MEMORY.
7716 042136 000000          GTBILO:  .WORD  0
7717 042140 062737 000002 042134  GTBIHI:  .WORD  0
7718 042146 005537 042136          ADD      #2,GTBILO
7719 042152 013700 042136          ADC      GTBIHI
7720 042156 013701 042134          MOV      GTBIHI,R0          ;COMPUTE THE 2K BLOCK SIZE OF MEMORY.
7721 042162 073027 177764          MOV      GTBILO,R1
7722 042166 010137 042220          ASHC     #-12,R0
7723 042172 162701 000011          MOV      R1,GTMSIZ
7724 042176 010137 042222          SUB      #11,R1
7725 042202 012737 123456 042224  MOV      R1,AVMBL
7726 042210 012737 123456 042226  MOV      #123456,GTRNL
7727 042216 000207          MOV      #123456,GTRNH
7728          RTS      PC
7729 042220 000000          GTMSIZ:  .WORD  0
7730 042222 000000          AVMBL:   .WORD  0
7731 042224 000000          GTRNL:   .WORD  0
7732 042226 000000          GTRNH:   .WORD  0
7733 042230 000000          GETMP1:  .WORD  0
7734
7735          ;THIS ROUTINE IS CALLED TO ALLOCATE A MEMORY BUFFER OF 2K WORDS LENGTH.
7736          ;IT IS CALLED USING A JSR PC INSTRUCTION FOLLOWED BY THE TABLE ENTRY
7737          ;OF RS4RB TO BE UPDATED.
7738 042232 000237          GETBUF:  SPL      7          ;LOCK OUT INTERRUPTS.
7739 042234 011637 042230          MOV      (SP),GETMP1
7740 042240 062716 000002          ADD      #2,(SP)          ;PICK UP A POINTER TO THE ARGUMENT
    
```



```

7741
7742 042244 104412
7743 042246 013737 042224 054372 1$: SAVREG
7744 042254 013737 042226 054370 MOV GTRNL,$LONUM
7745 042262 004737 054272 JSR GTRNH,$HINUM
7746 042266 013737 054372 042224 MOV PC,$RAND
7747 042274 013701 054370 MOV $LONUM,GTRNL
7748 042300 010137 042226 MOV $HINUM,R1
7749 042304 005000 CLR R1,GTRNH
7750 042306 071037 042222 DIV RO
7751
7752 042312 012702 036272 MOV #RS4RB,R2 ;AND UPDATE THE RETURN ADDRESS.
7753 042316 012703 000005 MOV #5,R3 ;SEE IF THIS AREA IS ALREADY IN USE.
7754 042322 062701 000011 ADD #11,R1
7755
7756 042326 020122 2$: CMP R1,(R2)+
7757 042330 001746 BEQ 1$ ;IF IT IS THEN TRY AGAIN.
7758 042332 077303 SOB R3,2$
7759
7760 042334 017704 177670 MOV @GETMP1,R4 ;OTHERWISE GIVE THIS BUFFER TO THE DRIVER.
7761 042340 010114 MOV R1,(R4)
7762 042342 104413 RESREG
7763 042344 000230 SPL 0
7764 042346 000207 RTS PC
7765
7766
7767 042350 104414 INDONE: RSET
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
7780
7781 042352 000004
7782
7783 042354 012737 050102 055572
7784
7785 042362 104414
7786 042364 113737 001502 001632
7787 042372 004737 057066
7788 042376 113737 057460 043112
7789 042404 113737 057461 043113
7790 042412 113737 057462 043114
7791
7792 042420 000137 043230 NN1: JMP NNDEV ;GO COMPUTE THE DRIVE NUMBERS.
7793
7794 042424 005037 043110 NN2: CLR NNGRPF ;FLAG WHICH DESIGNATES WHICH GOUP IS BEING
7795 042430 012737 000044 043106 MOV #S1MO,NNGRM ;TESTED ON THIS PASS.
7796 042436 012737 000030 043104 MOV #SOM1,NNGRS ;TEST GROUP ZERO FIRST.

```

```

*****
:*TEST 36 MASS BUS WRITE HIT CYCLE, INVALIDATION TEST
:*
:*THIS IS A TEST OF CACHE INVALIDATION ON MASS BUS CYCLES WHICH ARE
:*WRITE HITS IN THE CACHE. A GROUP OF LOCATIONS IS MADE HITS AND THEN A
:*MASS BUS DEVICE IS CALLED UPON TO DO TRANSFERS, WRITES TO THOSE
:*LOCATIONS. THOSE WRITES SHOULD THUS BE INVALIDATED.
:*
*****

```

TST36: SCOPE

```

;SET THE SKAD REGISTER
;IN CASE THE TEST ABORTS.

```

```

;DETERMINE WHAT DEVICES ARE AVAILABLE.

```

```

7797
7798 042444 004737 043120      NN3:  JSR    PC,NNSTUP      ;GO MAKE THE TEST ADDRESSES HITS
7799 042450 004777 000426      JSR    PC,@NNUD        ;USE THE FIRST DEVICE.
7800
7801
7802 042454 012700 140000      MOV    #TESTR1,R0
7803 042460 012701 000400      MOV    #256.,R1        ;MAKE SURE THOSE ADDRESSES ARE MISSES.
7804
7805 042464 005710                1$:   TST    (R0)
7806 042466 032737 000010 177752  BIT    #10,@#HITMIS
7807 042474 001430                BEQ    2$
7808
7809 042476 013737 043110 001634  MOV    NNGRPF,$TMP1    ;GOT A HIT REPORT FAILURE.
7810 042504 010037 001636      MOV    R0,$TMP2
7811 042510 005037 001640      CLR    $TMP3
7812 042514 023727 043102 042716  CMP    NNUD,#NNRS4     ;WAS THE RS4 DOING THE TRANSFER?
7813 042522 001003                BNE    11$             ;BRANCH IF NOT.
7814 042524 104151                ERROR  151
7815 042526 000137 042564      JMP    NN5
7816 042532 023727 043102 043010 11$:  CMP    NNUD,#NNRP4     ;WAS IT THE RP4?
7817 042540 001003                BNE    12$
7818 042542 104152                ERROR  152
7819 042544 000137 042564      JMP    NN5
7820 042550 104153                12$:  ERROR  153
7821 042552 000137 042564      JMP    NN5
7822
7823 042556 062700 000004      2$:   ADD    #4,R0
7824 042562 077140                SOB    R1,1$
7825
7826 042564 005237 043110      NN5:  INC    NNGRPF          ;TESTED BOTH GROUPS?
7827 042570 022737 000002 043110  CMP    #2,NNGRPF
7828 042576 001410                BEQ    NN6             ;BRANCH IF YES.
7829 042600 012737 000044 043104  MOV    #S1M0,NNGRS     ;IF NOT GO BACK AND TEST GROUP ONE.
7830 042606 012737 000030 043106  MOV    #SOM1,NNGRM
7831 042614 000137 042444      JMP    NN3
7832
7833 042620 000137 043476      NN6:  JMP    NNDONE
7834
7835 042624 104430      NNRH4: CALRH4          ;THIS IS THE CALL TO READ THE MASS BUS TESTER.
7836 042626          071      .BYTE  71
7837 042627          000      NNRH4U: .BYTE  0
7838 042630 052525      .WORD  52525
7839 042632 000000      .WORD  0
7840 042634 140000      .WORD  TESTR1
7841 042636 000000      .WORD  0
7842 042640 001000      .WORD  512.
7843 042642 042654      .WORD  2$
7844
7845 042644 005737 063500      1$:   TST    RH4ER1          ;ANY DEVICE ERRORS?
7846 042650 100401                BMI    2$             ;BRANCH IF YES.
7847 042652 000207                RTS    PC              ;IF NOT RETURN.
7848
7849 042654 013737 063502 001634 2$:   MOV    RH4ER2,$TMP1    ;REPORT DEVICE ERROR.
7850 042662 013737 063504 001636  MOV    RH4ER3,$TMP2
7851
7852 042670 013737 063506 001640  MOV    RH4ER4,$TMP3
    
```

```

7853 042676 005726          TST      (SP)+
7854 042700 104156          ERROR    156
7855 042702 105037 057462   CLRB    RH4DFL
7856 042706 105037 043114   CLRB    RH4FT
7857 042712 000137 042420   JMP     NN1
7858
7859 042716 104426          NNRS4:  CALRS4          ;THIS IS A CALL TO DO AN RS4 READ.
7860 042720      071          .BYTE   71
7861 042721      000          NNRS4U: .BYTE   0
7862 042722 000000          .WORD   0
7863 042724 000000          .WORD   0
7864 042726 140000          .WORD  TESTR1
7865 042730 000000          .WORD   0
7866 042732 001000          .WORD  512.
7867 042734 042746          .WORD   2$
7868
7869 042736 005737 061004          1$:    TST      RS4ER1          ;SEE IF THERE WERE DEVICE ERRORS.
7870 042742 100401          BMI     2$                ;BR IF YES.
7871 042744 000207          RTS     PC
7872
7873 042746 013737 061006 001634 2$:    MOV     RS4ER2,$TMP1
7874 042754 013737 061010 001636      MOV     RS4ER3,$TMP2
7875 042762 013737 061012 001640      MOV     RS4ER4,$TMP3
7876 042770 005726          TST     (SP)+
7877 042772 104154          ERROR    154
7878 042774 105037 057460   CLRB    RS4DFL
7879 043000 105037 043112   CLRB    RS4FT
7880 043004 000137 042420   JMP     NN1
7881
7882 043010 104427          NNRP4:  CALRP4          ;THIS IS A CALL TO DO AN RP4 READ.
7883 043012      071          .BYTE   71
7884 043013      000          NNRP4U: .BYTE   0
7885 043014 000000          .WORD   0
7886 043016 000000          .WORD   0
7887 043020 140000          .WORD  TESTR1
7888 043022 000000          .WORD   0
7889 043024 001000          .WORD  512.
7890 043026 043040          .WORD   2$
7891
7892 043030 005737 060034          1$:    TST      RP4ER1          ;WERE THERE ANY DEVICE ERRORS?
7893 043034 100401          BMI     2$
7894 043036 000207          RTS     PC
7895
7896 043040 013737 060036 001634 2$:    MOV     RP4ER2,$TMP1
7897 043046 013737 060040 001636      MOV     RP4ER3,$TMP2
7898 043054 013737 060042 001640      MOV     RP4ER4,$TMP3
7899 043062 005726          TST     (SP)+
7900 043064 104155          ERROR    155
7901 043066 105037 057461   CLRB    RP4DFL
7902 043072 105037 043113   CLRB    RP4FT
7903 043076 000137 042420   JMP     NN1
7904
7905 043102 000000          NNUD:   .WORD   0
7906
7907 043104 000000          NNGRS:  .WORD   0
7908 043106 000000          NNGRM:  .WORD   0
    
```

```

7909 043110 000000      NNGRPF: .WORD 0
7910
7911      ;THIS ROUTINE IS CALLED TO MAKE THE ADDRESSES IN TESTR1
7912      ;HITS PRIOR TO CALLING FOR THE MB DEVICE TO DO TRANSFERS.
7913 043112      000      RS4FT: .BYTE 0
7914 043113      000      RP4FT: .BYTE 0
7915 043114      000      RH4FT: .BYTE 0
7916 043115      000      RK5FT: .BYTE 0
7917 043116      000      UBEFT: .BYTE 0
7918      043120      .EVEN
7919
7920 043120 104412      NNSTUP: SAVREG
7921 043122 012700 043120      MOV      #NNSTUP,R0      ;MAKE THIS CODE HITS IN THE
7922 043126 012701 001000      MOV      #512.,R1      ;GROUP NOT BEING TESTED.
7923 043132 012702 142000      MOV      #TESTR2,R2
7924
7925 043136 013737 043106 177746 1$:      MOV      NNGRM,@#CONTRL
7926 043144 005720      TST      (R0)+
7927 043146 013737 043104 177746      MOV      NNGRS,@#CONTRL
7928 043154 005722      TST      (R2)+
7929 043156 077111      SOB      R1,1$
7930
7931 043160 013700 043104      2$:      MOV      NNGRS,R0
7932 043164 042700 000014      BIC      #14,R0
7933 043170 010037 177746      MOV      R0,@#CONTRL
7934 043174 012701 140000      MOV      #TESTR1,R1
7935 043200 012702 001000      MOV      #512.,R2
7936 043204 005721      3$:      TST      (R1)+
7937 043206 077202      SOB      R2,3$
7938 043210 013700 043106      MOV      NNGRM,R0
7939 043214 042700 000014      BIC      #14,R0
7940 043220 010037 177746      MOV      R0,@#CONTRL
7941 043224 104413      RESREG
7942 043226 000207      RTS      PC
7943
7944
7945      ;SEE WHAT DEVICE TO USE NEXT.
7946 043230 000240      NNDEV:  NOP
7947 043232 000240      NOP
7948 043234 005037 043102      CLR      NNUD
7949 043240 113700 043112      MOV      RS4FT,R0      ;IS THERE AN RS4 DRIVE.
7950 043244 001430      BEQ      NND2          ;BR IS NOT
7951
7952 043246 000240      NNDO:  NOP
7953 043250 012701 000001      MOV      #1,R1          ;FIND OUT WHAT DRIVE NUMBER IT IS.
7954 043254 012737 042716 043102      MOV      #NNRS4,NNUD
7955 043262 005002      CLR      R2
7956 043264 012703 000010      MOV      #10,R3
7957 043270 000240      1$:      NOP
7958 043272 030100      BIT      R1,R0
7959 043274 001406      BEQ      2$
7960 043276 140137 043112      BIC      R1,RS4FT      ;FOUND IT.
7961 043302 110237 042721      MOV      R2,NNRS4U
7962 043306 000137 042424      JMP      NN2
7963 043312 005202      2$:      INC      R2
7964 043314 006301      ASL      R1
    
```

```

7965 043316 077314 SOB R3,1$ ;KEEP LOOKING.
7966
7967 043320 104000 ERROR 0
7968 043322 105037 043112 CLRB RS4FT
7969
7970 043326 000240 NND2: NOP
7971 043330 113700 043113 MOVB RP4FT,R0 ;IS THERE AN RPO4 DRIVE.
7972 043334 001426 BEQ NND3 ;BR IF NO
7973 043336 012701 000001 MOV #1,R1
7974 043342 012737 043010 043102 MOV #NNRP4,NNUD
7975 043350 005002 CLR R2
7976 043352 012703 000010 MOV #10,R3
7977 043356 030100 1$: BIT R1,R0
7978 043360 001406 BEQ 2$
7979 043362 140137 043112 BICB R1,RS4FT
7980 043366 110237 043013 MOVB R2,NNRP4U
7981 043372 000137 042424 JMP NN2
7982 043376 005202 2$: INC R2
7983 043400 006301 ASL R1
7984 043402 077313 SOB R3,1$
7985 043404 104000 ERROR 0
7986 043406 105037 043113 CLRB RP4FT
7987
7988 043412 000240 NND3: NOP
7989 043414 113700 043114 MOVB RH4FT,R0 ;IS THERE A MASS BUS TESTER.
7990 043420 001426 BEQ NNDONE
7991 043422 012701 000001 MOV #1,R1
7992 043426 012737 042624 043102 MOV #NNRH4,NNUD
7993 043434 005002 CLR R2
7994 043436 012703 000010 MOV #10,R3
7995 043442 030100 1$: BIT R1,R0
7996 043444 001406 BEQ 2$
7997 043446 140137 043114 BICB R1,RH4FT
7998 043452 110237 042627 MOVB R2,NNRH4U
7999 043456 000137 042424 JMP NN2
8000 043462 005202 2$: INC R2
8001 043464 006301 ASL R1
8002 043466 077313 SOB R3,1$
8003 043470 104000 ERROR 0
8004 043472 105037 043114 CLRB RH4FT
8005 043476 104414 NNDONE: RSET
8006
8007
8008
8009 043500 105737 001750 TSTB KB11CM ;11/74 (KB11CM)?
8010 043504 001005 BNE 1$ ;BRANCH IF YES
8011 043506 105737 001747 TSTB KB11EM ;KB11-EM?
8012 043512 001002 BNE 1$ ;BR IF YES
8013 043514 000137 050102 JMP KT ;GO TO KT IF NO
8014 043520 1$: ;ENTER HERE IF KB11-E
8015
8016
8017
8018
8019
8020

```

```

*****
*TEST 37 CHECK IVSS, VSIU BITS
;THIS TEST CHECKS THAT THE IVSS AND VSIU BITS OF THE CACHE
;CONTROL REGISTER CAN BE SET AND CLEARED. VCIP IS ALSO
;CHECKED.

```

```

8021                                     ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8022                                     ;*****
8023 043520 000004                       TST37: SCOPE
8024 043522 005037 177746                CLR @#CONTRL
8025 043526 005737 177746                TST @#CONTRL
8026 043532 001404                        BEQ 1$
8027 043534 013737 177746 001562        MOV @#CONTRL,$REGO
8028 043542 104055                        ERROR 55 ;CCR COULD NOT BE CLEARED
8029
8030 043544 012737 040000 177746 1$:    MOV #IVSS,@#CONTRL
8031 043552 022737 040000 177746        CMP #IVSS,@#CONTRL
8032 043560 001404                        BEQ 2$
8033 043562 013737 177746 001562        MOV @#CONTRL,$REGO
8034 043570 104056                        ERROR 56 ;IVSS COULD NOT BE SET
8035
8036 043572 042737 040000 177746 2$:    BIC #IVSS,@#CONTRL
8037 043600 001404                        BEQ 3$
8038 043602 013737 177746 001562        MOV @#CONTRL,$REGO
8039 043610 104057                        ERROR 57 ;IVSS COULD NOT BE CLEARED
8040
8041 043612 012737 020000 177746 3$:    MOV #VSIU,@#CONTRL
8042 043620 032737 020000 177746        BIT #VSIU,@#CONTRL
8043 043626 001004                        BNE 4$
8044 043630 013737 177746 001562        MOV @#CONTRL,$REGO
8045 043636 104060                        ERROR 60 ;VSIU COULD NOT BE SET
8046
8047 043640 012700 000050 177746 4$:    MOV #50,R0 ;WAIT FOR VCIP TO CLEAR
8048 043644 032737 010000 177746        BIT #VCIP,@#CONTRL
8049 043652 001405                        BEQ 5$
8050 043654 077007                        SOB R0,4$
8051 043656 013737 177746 001562        MOV @#CONTRL,$REGO
8052 043664 104061                        ERROR 61 ;VCIP DID NOT CLEAR WITHIN SOME
8053                                     ;SOME TIME AFTER VSIU WAS SET
8054 043666 042737 020000 177746 5$:    BIC #VSIU,@#CONTRL
8055 043674 032737 020000 177746        BIT #VSIU,@#CONTRL
8056 043702 001404                        BEQ 6$
8057 043704 013737 177746 001562        MOV @#CONTRL,$REGO
8058 043712 104062                        ERROR 62 ;VSIU COULD NOT BE CLEARED
8059 043714 032737 010000 177746 6$:    BIT #VCIP,@#CONTRL
8060 043722 001374                        BNE 6$
8061                                     ;*****
8062                                     ;*TEST 40 CHECK VSIU BIT, WITH IVSS ALREADY SET
8063                                     ;THIS TEST CHECKS THAT THE 'VALID STORE IN USE' (VISU)
8064                                     ;BIT CAN BE SET AND CLEARED WHEN THE IVSS IS
8065                                     ;ALREADY SET.
8066                                     ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8067                                     ;*****
8068 043724 000004                       TST40: SCOPE
8069
8070 043726 012737 040000 177746        MOV #IVSS,@#CONTRL
8071 043734 032737 020000 177746        BIT #VSIU,@#CONTRL
8072 043742 001404                        BEQ 1$
8073 043744 013737 177746 001562        MOV @#CONTRL,$REGO
8074 043752 104062                        ERROR 62 ;VALID STORE IN USE, BIT 13,
8075                                     ;COULD NOT BE CLEARED IN CCR
8076 043754 032737 010000 177746 1$:    BIT #VCIP,@#CONTRL
    
```

```

8077 043762 001374          BNE      1$
8078 043764 052737 020000 177746  BIS      #VSIU,@#CONTRL
8079 043772 032737 020000 177746  BIT      #VSIU,@#CONTRL
8080 044000 001004          BNE      2$
8081 044002 013737 177746 001562  MOV      @#CONTRL,$REGO
8082 044010 104060          ERROR    60          ;VSIU (BIT 13) COULD NOT BE SET
8083                                     ;IN CCR (IVSS WAS ALREADY SET).
8084 044012 042737 020000 177746 2$:  BIC      #VSIU,@#CONTRL
8085 044020 032737 020000 177746  BIT      #VSIU,@#CONTRL
8086 044026 001404          BEQ      TST41          ;:EXIT
8087 044030 013737 177746 001562  MOV      @#CONTRL,$REGO
8088 044036 104062          ERROR    62          ;VSIU COULD NOT BE CLEARED IN CCR
8089                                     ;IVSS WAS ALREADY SET.
8090
8091 ::*****
8092 :*TEST 41      CHECK VCIP SETS WHEN CF IS SET
8093 :THIS TEST CHECKS THAT THE VCIP SETS WHEN CACHE-FLUSH IS
8094 :DONE AND IT CLEARS OUT WITHIN A CERTAIN TIME AFTER
8095 :THE FLUSH OF VALID STORE IS OVER
8096 :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8097 ::*****
8098 TST41:  SCOPE
8099 044040 000004          MOV      #FCAC, @# CONTRL;FLUSH CACHE
8100 044042 012737 000400 177746  NOP
8101 044050 000240          MOV      #50,RO
8102 044052 012700 000062          BIT      #VCIP, @# CONTRL
8103 044056 032737 010000 177746  BNE      1$
8104 044064 001004          MOV      @#CONTRL,$REGO
8105 044066 013737 177746 001562  ERROR    63          ;VCIP DID NOT SET WHEN CACHE
8106                                     ;FLUSH WAS ISSUED
8107 044076 032737 010000 177746 1$:  BIT      #VCIP, @#CONTRL ;WAIT FOR VCIP TO CLEAR
8108 044104 001405          BEQ      2$
8109 044106 077005          SOB      RO,1$
8110 044110 013737 177746 001562  MOV      @#CONTRL,$REGO
8111 044116 104061          ERROR    61          ;VCIP DID NOT CLEAR WITHIN A
8112                                     ;CERTAIN TIME AFTER CACHE FLUSH
8113                                     ;WAS DONE
8114 044120          2$:
8115
8116 ::*****
8117 :*TEST 42      CHECK CACHE FLUSH & VALID STORE SWITCHING
8118 :THIS TEST CHECKS THAT WHEN A CACHE FLUSH IS DONE
8119 :BY SETTING CF IN CCR, THE VALID STORE IN USE
8120 :(VSIU) SURTCHEs. VALID STORE SWITCHING FROM STORE-A
8121 :TO STORE-B AND VICE-VERSA IS CHECKED
8122 :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8123 ::*****
8124 TST42:  SCOPE
8125 044120 000004          CLR      @#CONTRL
8126 044122 005037 177746          BIT      #VCIP, @#CONTRL
8127 044126 032737 010000 177746 1$:  BNE      1$
8128 044134 001374          MOV      #FCAC, @#CONTRL ;FLUSH CACHE
8129 044136 012737 000400 177746  BIT      #VSIU, @#CONTRL
8130 044144 032737 020000 177746  BNE      2$
8131 044152 001004          MOV      @#CONTRL,$REGO
8132 044154 013737 177746 001562  ERROR    64          ;VSIU DID NOT SWITCH FROM 0 TO 1
8133 044162 104064

```

```

8133                                     :WHEN CACHE FLUSH WAS SET
8134 044164 032737 010000 177746 2$: BIT #VCIP, @#CONTRL
8135 044172 001374 BNE 2$
8136 044174 012737 000400 177746 MOV #FCAC, @#CONTRL
8137 044202 032737 020000 177746 BIT #VSIU, @#CONTRL
8138 044210 001404 BEQ 3$
8139 044212 013737 177746 001562 MOV @#CONTRL, $REGO
8140 044220 104064 ERROR 64 :VSIU DID NOT SWITCH FROM 1 TO 0 WHEN
8141                                     :FLUSH-CACHE WAS SET IN CCR
8142 044222 032737 010000 177746 3$: BIT #VCIP, @#CONTRL
8143 044230 001374 BNE 3$
8144
8145
8146
8147
8148
8149
8150
8151
8152
8153 044232 000004
8154 044234 005037 177746
8155 044240 032737 010000 177746 1$: BIT #VCIP, @#CONTRL
8156 044246 001374 BNE 1$
8157 044250 012737 040000 177746 MOV #IVSS, @#CONTRL ;SET IVSS
8158 044256 052737 000400 177746 BIS #FCAC, @#CONTRL ;FLUSH CACHE
8159 044264 032737 020000 177746 BIT #VSIU, @#CONTRL
8160 044272 001404 BEQ 2$ ;CHECK VSIU DID NOT SWITCH
8161 044274 013737 177746 001562 MOV @#CONTRL, $REGO
8162 044302 104065 ERROR 65 ;VSIU SWITCHED, WHEN IVSS
8163                                     :WAS SET AND CACHE FLUSH
8164                                     :WAS DONE, IT SHOULD NOT SWITCH
8165 044304 032737 010000 177746 2$: BIT #VCIP, @#CONTRL
8166 044312 001374 BNE 2$
8167 044314 052737 020000 177746 BIS #VSIU, @#CONTRL
8168 044322 032737 010000 177746 3$: BIT #VCIP, @#CONTRL
8169 044330 001374 BNE 3$
8170 044332 052737 000400 177746 BIS #FCAC, @#CONTRL
8171 044340 032737 020000 177746 BIT #VSIU, @#CONTRL ;CHECK VSIU DID NOT SWITCH
8172 044346 001004 BNE 4$
8173 044350 013737 177746 001562 MOV @#CONTRL, $REGO
8174 044356 104065 ERROR 65 ;VSIU SWITCHED, WHEN IVSS
8175                                     :WAS SET AND CACHE FLUSH WAS
8176                                     :DONE; IT SHOULD NOT SWITCH
8177 044360 032737 010000 177746 4$: BIT #VCIP, @#CONTRL
8178 044366 001374 BNE 4$
8179
8180
8181
8182
8183
8184
8185
8186
8187
8188
    
```

```

:*****
:*TEST 43 CHECK IVSS INHIBITS SWITCHING OF VALID STORE IN USE
:THIS TEST CHECKS THAT WHEN 'INHIBIT VALID STORE SWITCHING'
:(IVSS) IS SET AND FLUSH-CACHE BIT IS SET, THE
:VALID STORE IN USE DOES NOT SWITCH
:THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
:*****
    
```

```

TST43: SCOPE
CLR @#CONTRL
8155 044240 032737 010000 177746 1$: BIT #VCIP, @#CONTRL
8156 044246 001374 BNE 1$
8157 044250 012737 040000 177746 MOV #IVSS, @#CONTRL ;SET IVSS
8158 044256 052737 000400 177746 BIS #FCAC, @#CONTRL ;FLUSH CACHE
8159 044264 032737 020000 177746 BIT #VSIU, @#CONTRL
8160 044272 001404 BEQ 2$ ;CHECK VSIU DID NOT SWITCH
8161 044274 013737 177746 001562 MOV @#CONTRL, $REGO
8162 044302 104065 ERROR 65 ;VSIU SWITCHED, WHEN IVSS
8163                                     :WAS SET AND CACHE FLUSH
8164                                     :WAS DONE, IT SHOULD NOT SWITCH
8165 044304 032737 010000 177746 2$: BIT #VCIP, @#CONTRL
8166 044312 001374 BNE 2$
8167 044314 052737 020000 177746 BIS #VSIU, @#CONTRL
8168 044322 032737 010000 177746 3$: BIT #VCIP, @#CONTRL
8169 044330 001374 BNE 3$
8170 044332 052737 000400 177746 BIS #FCAC, @#CONTRL
8171 044340 032737 020000 177746 BIT #VSIU, @#CONTRL ;CHECK VSIU DID NOT SWITCH
8172 044346 001004 BNE 4$
8173 044350 013737 177746 001562 MOV @#CONTRL, $REGO
8174 044356 104065 ERROR 65 ;VSIU SWITCHED, WHEN IVSS
8175                                     :WAS SET AND CACHE FLUSH WAS
8176                                     :DONE; IT SHOULD NOT SWITCH
8177 044360 032737 010000 177746 4$: BIT #VCIP, @#CONTRL
8178 044366 001374 BNE 4$
    
```

```

:*****
:*TEST 44 CHECK VALID STORES (A & B) FOR GROUP 0
:THIS TEST CHECKS THE TWO VALID STORES (A&B) FOR GROUP 0
:OF THE CACHE. WHEN A CACHE-FLUSH IS ISSUED, THE CACHE
:SHOULD BE INVALIDATED BY SWITCHING THE VALID STORE
:IN USE
:THE TEST-CODE IS MADE HIT IN GROUP 1 (WHICH IS NOT
:BEING TESTED). THE TEST DATA IS MADE HIT IN GROUP 0.
    
```



8189 :FLUSH-CACHE BIT IS SET IN THE CCR. IT IS CHECKED THAT  
 8190 :THE TEST-DATA WHICH WAS HIT (MADE PREVIOUSLY) IN  
 8191 :GROUP 0 IS NO MORE A HIT. EACH LOCATION OF THE  
 8192 :TEST-DATA BLOCK IS REFERENCED AND CHECKED IF  
 8193 :IT WAS A MISS. OTHERWISE AN ERROR IS REPORTED. AS A  
 8194 :RESULT OF THE CACHE FLUSH THE VALID STORE SHOULD  
 8195 :HAVE SWITCHED FROM 0 TO 1. THEN THE VALID STORE  
 8196 :IS FORCED TO BE 0 AND THE TEST-DATA IS REFERENCED  
 8197 :AGAIN. IT IS CHECKED IF IT WAS A MISS.  
 8198 :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM  
 8199

```

*****
TST44: SCOPE
VSGO: CLR R5
VSGOA: MOV R5, @#CONTRL
      BIT #VCIP, @#CONTRL
      BNE -6
      MOV #SOMOM1,R2
      MOV #S1MOM1,R3
      BIS R5,R2
      BIS R5,R3
      MOV #VSGO, R0 ;MAKE TEST-CODE HIT IN
      MOV #1000, R1 ;GROUP 1
1$: MOV R2, @#CONTRL;FORCE REPLACE GROUP 0
      TST 2000(R2)
      MOV R3, @#CONTRL ;FORCE REPLACE GROUP 1
      TST (R0)+
      SOB R1, 1$
      MOV #TSTDAT, R0 ;MAKE TEST-DATA HIT IN
      MOV #1000, R1 ;GROUP 0
      MOV R3, @#CONTRL
      BIC #MOM1, @#CONTRL ;FORCE REPLACE GROUP 0
2$: TST (R0)+
      SOB R1, 2$
      BIC #S0, @#CONTRL
      BIS #S1, @#CONTRL ;FORCE REPLACE GROUP 1
      BIS #FCAC, @#CONTRL ;FLUSH CACHE
      MOV @#CONTRL, R4
      XOR R5, R4 ;CHECK IF VSIU COMPLEMENTED
      BIT #VSIU, R4
      BNE 3$
      MOV @#CONTRL, $REGO
      ERROR 64 ;VSIU DID NOT SWITCH WHEN
      ;CACHE-FLUSH WAS DONE
8232 044550 052737 000014 177746 3$: BIS #MOM1, @#CONTRL ;MAKE TEST-CODE HIT IN
8233 044556 012700 044372 MOV #VSGO, R0 ;GROUP 1
8234 044562 012701 001000 MOV #1000,R1
8235 044566 005720 4$: TST (R0)+
8236 044570 077102 SOB R1, 4$
8237 044572 042737 000014 177746 BIC #MOM1, @#CONTRL
8238 044600 012700 116310 MOV #TSTDAT, R0 ;REFERENCE TEST-DATA AND CHECK
8239 044604 012701 000400 MOV #400, R1 ;THAT IT IS A MISS. NOTE
8240 044610 005710 5$: TST (R0) ;SETTING CACHE-FLUSH SHOULD
8241 044612 032737 000010 177752 BIT #10, @#HITMIS ;HAVE INVALIDATED GROUP 0
8242 044620 001410 BEQ 6$
8243 044622 013737 177746 001562 MOV @#CONTRL, $REGO
8244 044630 005037 001564 CLR $REG1 ;GROUP NO.
    
```

```

8245 044634 010037 001566      MOV    R0,$REG2      ;TEST DATA ADDRESS
8246 044640 104066      ERROR  66           ;TEST-DATA WAS NOT A MISS.
8247                                     ;TEST DATA WAS MADE A HIT
8248                                     ;IN GROUP 0 AND THEN CACHE-
8249                                     ;FLUSH WAS DONE. CACHE-FLUSH
8250                                     ;SHOULD HAVE INVALIDATED GROUP
8251                                     ;0'S CACHED DATA, HENCE, THE
8252                                     ;TEST DATA REFERENCE SHOULD
8253                                     ;HAVE BEEN A MISS.
8254                                     ;PROBLE FAULURE
8255 044642 062700 000004      6$:  ADD    #4, R0      ;VALID STORE IS NOT BEING SWITCHED
8256 044646 077120      SOB    R1, 5$      ;TO THE OTHER WHEN CACHE-FLUSH IS
8257                                     ;SET IN THE CCR
8258 044650 032737 010000 177746 7$:  BIT    #VCIP, @#CONTRL
8259 044656 001374      BNE    7$
8260 044660 012700 020000      MOV    #VSIU, R0      ;COMPLEMENT VSIU
8261 044664 074037 177746      XOR    R0, @#CONTRL
8262 044670 032737 010000 177746 8$:  BIT    #VCIP, @#CONTRL
8263 044676 001374      BNE    8$
8264 044700 052737 000014 177746      BIS    #MOM1, @#CONTRL ;MAKE TEST-CODE HIT IN
8265 044706 012700 044372      MOV    #VSG0, R0      ;GROUP 1
8266 044712 012701 001000      MOV    #1000, R1
8267 044716 005720      9$:  TST    (R0)+
8268 044720 077102      SOB    R1, 9$
8269 044722 042737 000014 177746      BIC    #MOM1, @#CONTRL
8270                                     ;THE ORIGINAL VALID STORE (WHICH
8271                                     ;WAS INVALIDATED BY CACHE FLUSH)
8272                                     ;IS IN USE AGAIN.
8273 044730 012700 116310      MOV    #TSTDAT, R0
8274 044734 012701 000400      MOV    #400, R1      ;REFERENCE THE TEST-DATA AND
8275                                     ;CHECK IT IS A MISS
8276 044740 005710      10$: TST    (R0)
8277 044742 032737 000010 177752      BIT    #10, @#HITMIS
8278 044750 001410      BEQ    11$
8279 044752 013737 177746 001562      MOV    @#CONTRL,$REG0
8280 044760 005037 001564      CLR    $REG1          ;GROUP NO.
8281 044764 010037 001566      MOV    R0,$REG2      ;TEST DATA ADDRESS
8282 044770 104067      ERROR  67           ;TEST-DATA REFERENCE WAS NOT A MISS (IN
8283                                     ;GROUP 0, ORIGINAL VALID STORE). CACHE-FLUSH
8284                                     ;DONE EARLIER ON THE ORIGINAL VALID STORE
8285                                     ;SHOULD HAVE RESULTED IN INVALIDATING
8286                                     ;THE VALID STORE, THUS RESULTING IN
8287                                     ;CACHE-MISS ON TEST DATA REFERENCE.
8288                                     ;PROBALE FAILURE: VALID STORE IN USE IS NOT
8289                                     ;BEING INVALIDATD WHEN CACHE-FLUSH IS
8290                                     ;SET
8291 044772 062700 000004      11$: ADD    #4, R0
8292 044776 077120      SOB    R1, 10$
8293 045000 012701 020000      MOV    #VSIU,R1
8294 045004 074105      XOR    R1,R5          ;TESTED VALID STORE B (1)?
8295 045006 001402      BEQ    TST45          ;:EXIT
8296 045010 000137 044374      JMP    VSG0A
8297
8298
8299
8300
    ;*****
    ;*TEST 45      CHECK VALID STORES (A&B) FOR GROUPE 0 & 1
    ;*****
    
```

```

8301 ;THIS TEST CHECKS THAT HIT CAN BE OBTAINED FROM BOTH GROUPS
8302 ;(0&1) OF THE CACHE, FROM EACH OF THE TWO VALID
8303 ;STORES (A&B) PER GROUP. THUS ALL 4 VALID STORES GET
8304 ;CHECKED.
8305 ;TEST-DATA (UNIQUE) IS MADE A HIT IN GROUP 0 USING
8306 ;THE FIRST VALID STORE A. TEST-CODE IS MADE A HIT IN THE
8307 ;GROUP NOT BEING TESTED. TEST-DATA IS READ BACK AND
8308 ;CHECKED FOR CORRECTNESS. IT IS ALSO CHECKED IF THE
8309 ;TEST-DATA REFERENCE WAS A HIT. THE TESTING IS
8310 ;REPEATED FOR VALID STORE B.
8311 ;THE ENTIRE TEST (ABOVE) IS REPEATED FOR GROUP 1.
8312 ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8313 :*****
8314 045014 000004 TST45: SCOPE
8315 045016 005002 G1GOV: CLR R2 ;VSIU HIT MASK
8316 045020 005005 CLR R5 ;INITIALIZE COUNT DATA PATTERN TO BE USED
8317 045022 012700 000034 G1GOVA: MOV #SOMOM1, R0
8318 045026 012701 000054 MOV #S1MOM1, R1
8319 045032 010237 177746 G1GOVB: MOV R2, @#CONTRL
8320 045036 032737 010000 177746 1$: BIT #VCIP, @#CONTRL
8321 045044 001374 BNE 1$
8322 045046 050200 BIS R2, R0
8323 045050 050201 BIS R2, R1
8324 045052 012703 045016 MOV #G1GOV, R3 ;MAKE TEST-CODE HIT IN THE
8325 045056 012704 001000 MOV #1000, R4 ;GROUP NOT BEING TESTED
8326 045062 010037 177746 2$: MOV R0, @#CONTRL
8327 045066 005763 002000 TST 2000 (R3)
8328 045072 010137 177746 MOV R1, @#CONTRL
8329 045076 005723 TST (R3)+
8330 045100 077410 SOB R4, 2$
8331 045102 042700 000014 BIC #MOM1, R0 ;WRITE COUNT PATTERN AND MAKE
8332 045106 042701 000014 BIC #MOM1, R1 ;IT A HIT IN THE GROUP BEING
8333 045112 012703 116310 MOV #TSTDAT, R3 ;TESTED.
8334 045116 012704 001000 MOV #1000, R4 ;BIT 15 OF THE COUNT PATTERN INDICATES
8335 ;WHICH GROUP; BIT15=0, GROUP 0, ELSE 1
8336 045122 010037 177746 3$: MOV R0, @#CONTRL ;BIT 14 OF THE COUNT PATTERN INDICATES
8337 045126 010513 MOV R5, (R3) ;WHICH VALID STORE, A (0) OR B (1)
8338 045130 005723 TST (R3)+ ;MAKE IT A HIT
8339 045132 005205 INC R5
8340 045134 077404 SOB R4, 3$
8341 045136 010137 177746 MOV R1, @#CONTRL
8342 045142 012703 116310 MOV #TSTDAT, R3
8343 045146 012704 001000 MOV #1000, R4
8344 045152 042705 001777 BIC #1777, R5 ;INITIALIZE PATTERN TO BE CHECKED
8345 045156 011337 045332 4$: MOV (R3), TMP ;READ THE TEST-DATA AND
8346 045162 032737 000020 177752 BIT #20, @#HITMIS ;CHECK IF THE REFERENCE WAS
8347 045170 001016 BNE 5$ ;A HIT
8348 045172 013737 177746 001562 MOV @#CONTRL, $REG0
8349 045200 005037 001564 CLR $REG1 ;GROUP NO.
8350 045204 032705 100000 BIT #BIT15, R5 ;WHICH GROUP?
8351 045210 001403 BEQ 8$
8352 045212 012737 000001 001564 8$: MOV #1, $REG1
8353 045220 010337 001566 MOV R3, $REG2 ;TEST DATA ADDRESS
8354 045224 104070 ERROR 70 ;TEST-DATA REFERENCE WAS NOT A
8355 ;HIT, FROM THE GROUP AND
8356 ;VALID STORE BEING TESTED
    
```

```

8357 045226 023705 045332 5$: CMP TMP, R5 ;DATA CORRECT?
8358 045232 001410 BEQ 6$
8359 045234 010537 001562 MOV R5, $REG0 ;EXPCTD DATA
8360 045240 013737 045332 001564 MOV TMP, $REG1 ;DATA RECVD
8361 045246 010337 001566 MOV R3, $REG2
8362 045252 104071 ERROR 71 ;READ INCORRECT DATA ON REFEREN
8363 ; -CING A CACHED LOCATION.
8364 045254 062703 000002 6$: ADD #2, R3
8365 045260 005205 INC R5
8366 045262 077443 SOB R4, 4$
8367 045264 012704 020000 MOV #VSIU, R4
8368 045270 074402 XOR R4, R2 ;DONE VALID STORE B (1)?
8369 045272 001405 BEQ 7$ ;YES
8370 045274 052705 040000 BIS #BIT14, R5 ;INDICATE VS-B IN DATA-PATTERN
8371 045300 042705 001777 BIC #1777, R5
8372 045304 000646 BR G1GOVA ;CHECK GROUP, VS-B
8373 045306 032705 100000 7$: BIT #BIT15, R5 ;DONE CHECKING GROUP 1?
8374 045312 001010 BNE TST46 ;:EXIT
8375 045314 012700 000054 MOV #S1MOM1, R0
8376 045320 012701 000034 MOV #SOMOM1, R1
8377 045324 012705 100000 MOV #BIT15, R5 ;INDICATE GROUP 1
8378 045330 000640 BR G1GOVB
8379 045332 000000 TMP: .WORD 0
    
```

```

8380
8381
8382 ::*****
8383 :*TEST 46 CHECK VALID STORES (A & B ) FOR GROUP 1
8384 ;THIS TEST CHECKS RTHE TWO VALID STORES (A&B) FOR GROUP 1
8385 ;OF THE CACHE. WHEN A CACHE-FLUSH IS ISSUED, THE CACHE
8386 ;SHOULD BE INVALIDATED BY SWITCHING THE VALID STORE
8387 ;IN USE.
8388 ;THE TEST-CODE IS MADE HIT IN GROUP 1 (WHICH IS NOT
8389 ;BEING TESTED). THE TEST DATA IS MADE HIT IN GROUP 0.
8390 ;FLUSH-CACHE HIT IS SET IN THE CCR. IT IS CHECKED THAT
8391 ;THE TEST-DATA WHICH WAS HIT (MADE PREVIOUSLY) IN
8392 ;GROUP 0 IS NO MORE A HIT, EACH LOCATION OF THE
8393 ;TEST-DATA BLOCK IS REFERENCED AND CHECKED IF
8394 ;IT WAS A MISS. OTHERWISE AN ERROR IS REPORTED. AS A
8395 ;RESULT OF THE CACHE FLUSH THE VALID STORE SHOULD
8396 ;HAVE SWITCHED FROM 0 TO 1. THEN THE VALID STORE
8397 ;IS FORCED TO BE 0 AND THE TEST-DATA IS REFERENCED
8398 ;AGAIN. IT IS CHECKED IF IT WAS A MISS.
8399 ;THE WHOLE TEST IS REPEATED USING VALID-STORE
8400 ;B (1).
8401 ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8402 ::*****
    
```

```

8403 045334 000004 TST46: SCOPE
8404 045336 005005 VSG1: CLR R5 ;R5, HIT MASK FOR VSIU
8405 045340 010537 177746 VSG1A: MOV R5, @#CONTRL
8406 045344 032737 010000 177746 BIT #VCIP, @#CONTRL
8407 045352 001374 BNE -6
8408 045354 012702 000034 MOV #SOMOM1, R2
8409 045360 012703 000054 MOV #S1MOM1, R3
8410 045364 050502 BIS R5, R2
8411 045366 050503 BIS R5, R3
8412 045370 012700 045336 MOV #VSG1, R0 ;MAKE TEST-CODE HIT IN
    
```

8413	045374	012701	001000			MOV	#1000, R1	:GROUP 0
8414	045400	010337	177746	1\$:		MOV	R3, @#CONTRL	:FORCE REPLACE GROUP 1
8415	045404	005760	002000			TST	2000(R0)	
8416	045410	010237	177746			MOV	R2, @#CONTRL	:FORCE REPLACE GROUP 0
8417	045414	005720				TST	(R0)+	
8418	045416	077110				SOB	R1, 1\$	
8419	045420	012700	116310			MOV	#TSTDAT, R0	:MAKE TEST-DATA HIT IN
8420	045424	012701	001000			MOV	#1000, R1	:GROUP 1
8421	045430	010337	177746			MOV	R3, @#CONTRL	:FORCE REPLACE GROUP 1
8422	045434	042737	000014	177746		BIC	#MOM1, @#CONTRL	
8423	045442	005720			2\$:	TST	(R0)+	
8424	045444	077102				SOB	R1, 2\$	
8425	045446	042737	000040	177746		BIC	#S1, @#CONTRL	
8426	045454	052737	000020	177746		BIS	#S0, @#CONTRL	:FORCE REPLACE GROUP 0
8427	045462	052737	000400	177746		BIS	#FCAC, @#CONTRL	:FLUSH CACHE
8428	045470	013704	177746			MOV	@#CONTRL, R4	
8429	045474	074504				XOR	R5, R4	:CHECK IF VSIU COMPLEMENTED
8430	045476	032704	020000			BIT	#VSIU, R4	
8431	045502	001004				BNE	3\$	
8432	045504	013737	177746	001562		MOV	@#CONTRL, \$REGO	
8433	045512	104064				ERROR	64	:VSIU DID NOT SWITCH WHEN
8434								:CACHE-FLUSH WAS DONE
8435	045514	052737	000014	177746	3\$:	BIS	#MOM1, @#CONTRL	:MAKE TEST-CODE HIT IN
8436	045522	012700	045336			MOV	#VSG1, R0	:GROUP 0
8437	045526	012701	001000			MOV	#1000, R1	
8438	045532	005720			4\$:	TST	(R0)+	
8439	045534	077102				SOB	R1, 4\$	
8440	045536	042737	000014	177746		BIC	#MOM1, @#CONTRL	
8441	045544	012700	116310			MOV	#TSTDAT, R0	:REFERENCE TEST-DATA AND CHECK
8442	045550	012701	000400			MOV	#400, R1	:THAT IT IS A MISS. NOTE
8443	045554	005710			5\$:	TST	(R0)	:SETTING CACHE-FLUSH SHOULD
8444	045556	032737	000010	177752		BIT	#10, @#HITMIS	:HAVE INVALIDATED GROUP
8445	045564	001411				BEQ	6\$	
8446	045566	013737	177746	001562		MOV	@#CONTRL, \$REGO	
8447	045574	012737	000001	001564		MOV	#1, \$REG1	:GROUP NO.
8448	045602	010037	001566			MOV	R0, \$REG2	:TEST DATA ADDRESS
8449	045606	104066				ERROR	66	:TEST-DATA WAS NOT A MISS.
8450								:TEST DATA WAS MADE A HIT
8451								:IN GROUP 1 AND THEN CACHE-
8452								:FLUSH WAS DONE. CACHE-FLUSH
8453								:SHOULD HAVE INVALIDATED GROUP
8454								:1'S CACHED DATA. HENCE, THE
8455								:TEST DATA REFERENCE SHOULD
8456								:HAVE BEEN A MISS.
8457								:PROBABLE FAILURE:
8458	045610	062700	000004		6\$:	ADD	#4, R0	:VALID STORE IS NOT BEING SWITCHED
8459	045614	077121				SOB	R1, 5\$	:TO THE OTHER WHEN CACHE-FLUSH IS
8460								:SET IN THE CCR
8461	045616	032737	010000	177746	7\$:	BIT	#VCIP, @#CONTRL	
8462	045624	001374				BNE	7\$	
8463	045626	012700	020000			MOV	#VSIU, R0	:COMPLEMENT VSIU
8464	045632	074037	177746			XOR	R0, @#CONTRL	
8465	045636	032737	010000	177746	8\$:	BIT	#VCIP, @#CONTRL	
8466	045644	001374				BNE	8\$	
8467	045646	052737	000014	177746		BIS	#MOM1, @#CONTRL	:MAKE TEST-CODE HIT IN
8468	045654	012700	045336			MOV	#VSG1, R0	:GROUP 0

```

8469 045660 012701 001000          MOV    #1000, R1
8470 045664 005720          9$:   TST    (R0)+
8471 045666 077102          SOB    R1, 9$
8472 045670 042737 000014 177746  BIC    #MOM1, @#CONTRL
8473                                     ;THE ORIGINAL VALID STORE (WHICH
8474                                     ;WAS INVALIDATED BY CACHE FLUSH)
8475                                     ;IS IN USE AGAIN.
8476 045676 012700 116310          MOV    #TSTDAT, R0
8477 045702 012701 000400          MOV    #400, R1
8478                                     ;REFERENCE THE TEST-DATA AND
8479                                     ;CHECK IT IS A MISS
8480 045706 005710          10$:  TST    (R0)
8481 045710 032737 000010 177752  BIT    #10, @#HITMIS
8482 045716 001411          BEQ    11$
8483 045720 013737 177746 001562  MOV    @#CONTRL, $REG0
8484 045726 012737 000001 001564  MOV    #1, $REG1
8485 045734 010037 001566          MOV    R0, $REG2
8486                                     ;GROUP NO.
8487                                     ;TEST DATA ADDRESS
8488                                     ;TEST-DATA REFERENCE WAS NOT A MISS (IN
8489                                     ;GROUP 1, ORIGINAL VALID STORE). CACHE-FLUSH
8490                                     ;DONE EARLIER ON THE ORIGINAL VALID STORE
8491                                     ;SHOULD HAVE RESULTED IN INVALIDATING
8492                                     ;THE VALID STORE, THUS RESULTING IN
8493                                     ;CACHE-MISS ON TEST DATA REFERENCE.
8494                                     ;PROBABLE FAILURE: VALID STORE IN USE IS NOT
8495                                     ;BEING INVALIDATED WHEN CACHE-FLUSH IS
8496                                     ;SET
8497 045742 062700 000004          11$:  ADD    #4, R0
8498 045746 077121          SOB    R1, 10$
8499 045750 012701 020000          MOV    #VSIU, R1
8500 045754 074105          XOR    R1, R5 ;TESTED VALID STORE B (1)?
8501 045756 001402          BEQ    TST47
8502 045760 000137 045340          JMP    VSG1A
8503
8504
8505
8506
8507
8508
8509
8510
8511
8512
8513

```

```

*****
*TEST 47 CHECK CACHE TURNS OFF WHEN FLUSH IS DONE WITH IVSS SET
;THIS TEST CHECKS THAT IF CACHE-FLUSH IS DONE (SETTING CF), WHEN IVSS
;IS SET, THE VALID STORES ARE NOT SWITCHED AND THE CACHE IS TURNED
;OFF (AND A SLOW FLUSH IS PERFORMED). THUS, ANY REFERENCE TO
;A PREVIOUSLY CACHED DATA SHOULD RESULT IN CACHE MISS.
;TEST-DATA IS MADE HIT IN GROUP 0 (BEING TESTED). TEST CODE IS
;MADE HIT IN GROUP 1. IVSS IS SET AND A FLUSH IS DONE. PREVIOUSLY
;CACHED TEST-DATA IS REFERENCED TO CHECK IT IS A MISS.
;THE TEST IS REPEATED FOR BOTH GROUPS AND VALID STORES.
;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
*****

```

```

8514 045764 000004          TST47: SCOPE
8515
8516 045766 005002          IVFC:  CLR    R2 ;BIT MASK FOR VSIU
8517 045770 012700 000034          MOV    #SOMOM1, R0
8518 045774 012701 000054          MOV    #S1MOM1, R1
8519 046000 050200          IVFCA: BIS    R2, R0
8520 046002 050201          BIS    R2, R1
8521 046004 010237 177746          MOV    R2, @#CONTRL
8522 046010 032737 010000 177746  1$:   BIT    #VCIP, @#CONTRL
8523 046016 001374          BNE    1$
8524

```

```

8525 046020 012703 045766      MOV      #IVFC,R3      ;MAKE TEST CODE BIT IN GROUP
8526 046024 012704 001000      MOV      #1000,R4     ;NOT BEING TESTED
8527 046030 010037 177746      MOV      R0,@#CONTRL  2$:
8528 046034 005763 002000      TST      2000(R3)
8529 046040 010137 177746      MOV      R1,@#CONTRL
8530 046044 005723      TST      (R3)+
8531 046046 077410      SOB      R4,2$
8532
8533 046050 042700 000014      BIC      #MOM1,R0
8534 046054 042701 000014      BIC      #MOM1,R1
8535
8536 046060 012703 116310      MOV      #TSTDAT,R3   ;MAKE TEST-DATA HIT IN GROUP
8537 046064 012704 001000      MOV      #1000,R4     ;BEING TESTED
8538
8539 046070 010037 177746      MOV      R0,@#CONTRL
8540 046074 005723      TST      (R3)+      3$:
8541 046076 077402      SOB      R4,3$
8542
8543 046100 010137 177746      MOV      R1,@#CONTRL   ;FORCE REPLACE GROUP (NOT BEING TESTED)
8544 046104 052737 040000 177746      BIS      #IVSS,@#CONTRL ;SET IVSS
8545
8546 046112 012705 000004      MOV      #4,R5        ;BIT MASK FOR HIT/MISS REGISTER
8547 046116 012704 000400      MOV      #400,R4
8548 046122 012703 120310      MOV      #TSTDAT+2000,R3
8549 046126 052737 000400 177746      BIS      #FCAC,@#CONTRL ;FLUSH CACHE
8550 046134 005743      TST      -(R3)        ;REFERENCE PREVIOUSLY CACHED
8551 046136 030537 177752      BIT      R5,@#HITMIS  ;TEST DATA& CHECK IT IS A MISS
8552 046142 001004      BNE      6$
8553 046144 162703 000002      SUB      #2,R3        5$:
8554 046150 077407      SOB      R4,4$
8555 046152 000417      BR       7$          ;DONE
8556
8557 046154 013737 177746 001562 6$:      MOV      @#CONTRL,$REG0
8558 046162 005037 001564      CLR      $REG1        ;GROUP NO.
8559 046166 032700 000040      BIT      #S1,R0       ;WHICH GROUP?
8560 046172 001403      BEQ      12$
8561 046174 012737 000001 001564      MOV      #1,$REG1    ;GROUP NO
8562 046202 010337 001566 12$:      MOV      R3,$REG2    ;TEST DATA ADDRESS
8563 046206 104072      ERROR    72          ;TEST DATA REFERENCE DID NOT
8564                                     ;REGISTER A MISS. TEST-DATA WAS
8565 046210 000755      BR       5$          ;MADE BIT IN A GROUP. CACHE-FLUSH
8566                                     ;WAS DONE, WITH IVSS SET. REFERENCE
8567                                     ;TO THE PREVIOUSLY CACHED DATA
8568                                     ;SHOULD HAVE BEEN A MISS.
8569                                     ;PROBABLE FAILURE: CACHE DOES NOT
8570                                     ;TURN OFF WHEN IVSS IS SET AND
8571                                     ;FLUSH IS PERFORMED
8572                                     ;CHECK THAT THE CACHE HAS TURNED ON AGAIN,CHECK
8573                                     ;THAT HITS CAN BE OBTAINED
8574
8575 046212 012703 045766      7$:      MOV      #IVFC,R3     ;MAKE THE TEST-CODE HIT IN GROUP NOT
8576 046216 012704 001000      MOV      #1000,R4     ;BEING TESTED
8577 046222 052700 000014      BIS      #MOM1,R0
8578 046226 052701 000014      BIS      #MOM1,R1
8579 046232 010037 177746      8$:      MOV      R0,@#CONTRL
8580 046236 005763 002000      TST      2000(R3)

```

```

8581
8582 046242 010137 177746      MOV      R1,@#CONTRL
8583 046246 005723              TST      (R3)+
8584 046250 077410              SOB      R4,8$
8585
8586 046252 042700 000014      BIC      #MOM1,R0
8587 046256 042701 000014      BIC      #MOM1,R1
8588
8589 046262 012703 116310      MOV      #TSTDAT,R3      ;MAKE TEST-DATA HIT IN GROUP
8590 046266 012704 001000      MOV      #1000,R4        ;BEING TESTED
8591
8592 046272 010037 177746      MOV      R0,@#CONTRL
8593 046276 005723      9$:      TST      (R3)+
8594 046300 077402              SOB      R4,9$
8595
8596
8597 046302 010137 177746      MOV      R1,@#CONTRL      ;FORCE REPLACE GROUP NOT BEING
8598 046306 012703 116310      MOV      #TSTDAT,R3      ;TESTED
8599 046312 012704 001000      MOV      #1000,R4        ;REFERENCE TEST-DATA (IN THE
8600 046316 005713              MOV      #1000,R4        ;GROUP BEING CHECKED) AND
8601 046320 032737 000010 177752 10$:      TST      (R3)            ;MAKE SURE IT IS A HIT
8602 046326 001016              BIT      #10,@#HITMIS    ;HIT?
8603 046330 013737 177746 001562      BNE     11$              ;YES
8604 046336 005037 001564      MOV      @#CONTRL,$REGO
8605 046342 032700 000040      CLR     $REG1            ;GROUP NO.
8606 046346 001403              BIT      #S1,R0          ;WHICH GROUP?
8607 046350 012737 000001 001564      BEQ     13$
8608 046356 010337 001566      MOV      #1,$REG1        ;GROUP NO
8609 046362 104073              MOV      R3,$REG2        ;TEST DATA ADDRESS
8610
8611
8612
8613
8614
8615
8616
8617 046364 062703 000002      11$:      ADD      #2,R3            ;PREVIOUSLY CACHED TEST-DATA
8618 046370 077426              SOB      R4,10$          ;WAS REFERENCED BUT IT
8619 046372 052700 000014      BIS     #MOM1,R0         ;WAS NOT A HIT.
8620 046376 052701 000014      BIS     #MOM1,R1         ;POSSIBLE FAULT: CACHE DID NOT
8621
8622 046402 012704 020000      MOV      #VSIU,R4        ;TURN ON AFTER HAVING TURNED
8623 046406 074402              XOR      R4,R2           ;OFF (WHEN A CACHE FLUSH
8624 046410 001402              BEQ     14$              ;WAS DONE WITH IVSS SET).
8625 046412 000137 046000      JMP     IVFCA
8626 046416 032700 000040      14$:      BIT      #S1,R0          ;YES, DONE GROUP 1?
8627 046422 001007              BNE     TST50            ;:YES,EXIT
8628
8629 046424 012700 000054      MOV      #S1MOM1,R0      ;CCR MASKS FOR GROUP 1 TESTING
8630 046430 012701 000034      MOV      #S0MOM1,R1
8631 046434 005002              CLR     R2               ;BIT MASK FOR VALID STORE
8632 046436 000137 046000      JMP     IVFCA            ;GO TEST GROUP 1
8633
8634
8635
8636

```

```

:*****
:*TEST 50      CHECK CACHE TURNS OFF ON A BACK-TO-BACK FLUSH

```



```

8637 :THIS TEST CHECKS THAT THE CACHE TURNS OFF AND FORCES
8638 :ALL REFERENCES TO THE MAIN MEMORY WHEN BACK-TO-BACK
8639 :CACHE FLUSHES ARE DONE. WHEN A CACHE FLUSH IS INITIATED
8640 :WHILE THE PREVIOUS ONE IS IN PROGRESS, IT IS KNOWN
8641 :AS BACK-TO-BACK FLUSH.
8642 :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8643 :*****
    
```

```

8644 046442 000004 TST50: SCOPE
8645 046444 005037 177746 CLR @#CONTRL
8646 046450 032737 010000 177746 1$: BIT #VCIP,@#CONTRL
8647 046456 001374 BNE 1$
8648 046460 012701 177774 MOV #-4,R1
8649 046464 012700 000007 MOV #7,R0
8650 046470 012737 000400 177746 MOV #FCAC,@#CONTRL ;FLUSH CACHE
8651 046476 012737 000400 177746 MOV #FCAC,@#CONTRL ;AGAIN FLUSH THE CACHE. SINCE
8652 ;PREVIOUS FLUSH IS STILL IN
8653 046504 005201 2$: INC R1 ;PROGRESS CACHE SHOULD BE
8654 046506 001410 BEQ 3$ ;TURNED OFF
8655 046510 030037 177752 BIT R0,@#HITMIS ;CHECK THAT THE LAST THREE REFERENCES
8656 046514 001773 BEQ 2$ ;WERE MISSES
8657 046516 013702 177752 MOV @#HITMIS,R2
8658 046522 010237 001562 MOV R2,$REGO
8659 046526 104115 ERROR 115 ;CACHE DID NOT TURN OFF ON
8660 ;PERFORMING A BACK-TO-BACK
8661 ;FLUSH. FOR THE PERIOD OF TIME
8662 ;THAT THE FLUSH IS BEING DONE
8663 ;AND THE CACHE IS OFF, ALL
8664 ;REFERENCES SHOULD BE FORCED
8665 ;TO MAIN MEMORY (MISSES).
8666 046530 3$: ;EXIT
8667
8668
8669 :*****
    
```

```

8670 :*TEST 51 CHECK CACHE-BYPASS
8671 :THIS TEST CHECKS THE CACHE BYPASS FUNCTION. WHEN THE
8672 :'BYPASS CACHE' IS SET IN THE CACHE CONTROL REGISTER
8673 :ALL REFERENCES ARE FORCED TO MAIN MEMORY. IF A
8674 :READ OR WRITE HIT OCCURS THAT LOCATION IS INVAL-
8675 :-IDATED IN THE TAG STORE.
8676 :FIRST, THE TEST CODE IS MADE HIT IN GROUP 1 BY
8677 :FORCE-REPLACING GROUP 1. THEN THE TEST-DATA IS MADE
8678 :HIT IN GROUP 0. CACHE-BYPASS IS SET AND THE TEST
8679 :DATA (WHICH HAS BEEN CACHED IN GROUP 0) IS
8680 :REFERENCED. THE REFERENCES ARE CHECKED FOR MISSES
8681 :(THE TEST-DATA INSIDE THE CACHE GROUP-0 SHOULD
8682 :HAVE BEEN INVALIDATED WHEN REFERENCES WERE
8683 :MADE WITH CACHE-BYPASS SET.)
8684 :THE ENTIRE TEST IS REPEATED, SELECTING THE
8685 :OTHER VALID STORE AND THEN WITH TEST-DATA IN
8686 :GROUP 1.
8687 :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8688 :*****
    
```

```

8689 046530 000004 TST51: SCOPE
8690 046532 005002 CBP: CLR R2 ;BIT MASK FOR VSIU
8691 046534 012700 000034 MOV #SOMOM1,R0
8692 046540 012701 000054 MOV #SIMOM1,R1
    
```

8693	046544	050200			CBPA:	BIS	R2,R0	
8694	046546	050201				BIS	R2,R1	
8695	046550	010237	177746			MOV	R2,@#CONTRL	
8696	046554	032737	010000	177746	1\$:	BIT	#VCIP,@#CONTRL	:SELECT VSIU
8697	046562	001374				BNE	1\$	
8698								
8699	046564	012703	046532			MOV	#CBP,R3	:MAKE TEST-CODE HIT IN THE
8700	046570	012704	001000			MOV	#1000,R4	:GROUP NOT BEING TESTED
8701	046574	010037	177746		2\$:	MOV	R0,@#CONTRL	
8702	046600	005763	002000			TST	2000(R3)	
8703	046604	010137	177746			MOV	R1,@#CONTRL	
8704	046610	005723				TST	(R3)+	
8705	046612	077410				SOB	R4,2\$	
8706								
8707	046614	042700	000014			BIC	#MOM1,R0	
8708	046620	042701	000014			BIC	#MOM1,R1	
8709								
8710	046624	012703	116310			MOV	#TSTDAT,R3	:MAKE TEST-DATA HIT IN THE
8711	046630	012704	001000			MOV	#1000,R4	:GROUP BEING TESTED
8712	046634	010037	177746			MOV	R0,@#CONTRL	
8713	046640	005723			3\$:	TST	(R3)+	
8714	046642	077402				SOB	R4,3\$	
8715	046644	010137	177746			MOV	R1,@#CONTRL	:FORCE REPLACE IN THE GROUP NOT
8716								:BEING TESTED
8717	046650	052737	001000	177746		BIS	#UCB,@#CONTRL	:UNCONDITIONED CACHE BY-PASS
8718								
8719	046656	012703	116310			MOV	#TSTDAT,R3	
8720	046662	012704	001000			MOV	#1000,R4	:REFERENCE THE CACHED-TEST-DATA
8721	046666	005713			4\$:	TST	(R3)	:THE GROUP BEING TESTED
8722	046670	032737	000010	177752		BIT	#10,@#HITMIS	:MISS?
8723	046676	001416				BEQ	5\$	:YES
8724								
8725	046700	013737	177746	001562		MOV	@#CONTRL,\$REG0	
8726	046706	005037	001564			CLR	\$REG1	:GROUP NO.
8727	046712	032700	000040			BIT	#S1,R0	:WHICH GROUP?
8728	046716	001403				BEQ	8\$	
8729	046720	012737	000001	001564		MOV	#1,\$REG1	:GROUP NO
8730	046726	010337	001566		8\$:	MOV	R3,\$REG2	:TEST DATA ADDRESS
8731	046732	104074				ERROR	74	:TEST-DATA-REFERENCE WAS NOT
8732								:A MISS. TEST-DATA WAS PREVIOUSLY
8733								:CACHED IN THE GROUP BEING
8734								:TESTED. THEN IT WAS REFERENCED
8735								:WITH CACHE BY-PASS SET. IT
8736								:SHOULD HAVE BEEN A MISS.
8737								:PROBABLE FAILURE : A MISS IS
8738								:NOT BEING FORCED WHEN CACHE
8739								:BYPASS IS SET
8740								
8741	046734	062703	000002		5\$:	ADD	#2,R3	
8742	046740	077426				SOB	R4,4\$	:DONE?
8743								
8744	046742	042737	001000	177746		BIC	#UCB,@#CONTRL	:CLEAR CACHE BYPASS
8745	046750	012703	116310			MOV	#TSTDAT,R3	:REFERENCE THE TEST-DATA AGAIN
8746	046754	012704	000400			MOV	#400,R4	:IT SHOULD BE A MISS
8747	046760	005713			6\$:	TST	(R3)	
8748	046762	032737	000010	177752		BIT	#10,@#HITMIS	:MISS?

```

8749 046770 001416 BEQ 7$ ;YES
8750
8751 046772 013737 177746 001562 MOV @#CONTRL,$REGO
8752 047000 005037 001564 CLR $REG1 ;GROUP NO.
8753 047004 032700 000040 BIT #S1,R0 ;WHICH GROUP?
8754 047010 001403 BEQ 9$
8755 047012 012737 000001 001564 MOV #1,$REG1 ;GROUP NO
8756 047020 010337 001566 9$: MOV R3,$REG2 ;TEST DATA ADDRESS
8757 047024 104075 ERROR 75 ;TEST-DATA-REFERENCE WAS NOT
8758 ;A MISS. TEST-DATA WAS PREVIOUSLY
8759 ;CACHED IN THE GROUP BEING
8760 ;TESTED. THEN IT WAS INVALIDATED
8761 ;BY REFERENCING IT WHILE
8762 ;CACHE-BYPASS WAS SET. THEN
8763 ;CACHE-BYPASS WAS CLEARED AND
8764 ;THE TEST DATA WAS REFERENCED
8765 ;AGAIN TO MAKE SURE IT WAS
8766 ;INVALIDATED.
8767 047026 062703 000004 7$: ADD #4,R3 ;PROBABLE FAILURE - CACHE-BYPASSS
8768 047032 077426 SOB R4,6$ ;DOES NOT INVALIDATE DATE
8769 ;THAT IS A HIT INSIDE THE
8770 047034 052700 000014 BIS #MOM1,R0 ;CACHE
8771 047040 052701 000014 BIS #MOM1,R1
8772 047044 012704 020000 MOV #VSIU,R4
8773 047050 074402 XOR R4,R2 ;DONE BOTH VALID STORES?
8774 047052 001234 BNE CBPA ;NO
8775 047054 032700 000040 BIT #S1,R0 ;TESTED GROUP 1
8776 047060 001005 BNE TST52 ;:EXIT
8777
8778 047062 012700 000054 MOV #S1MOM1,R0 ;SET UP FOR TESTING GROUP 1
8779 047066 012701 000034 MOV #SOMOM1,R1
8780 047072 000624 BR CBPA

```

```

8781
8782
8783
8784
8785
8786
8787
8788
8789
8790
8791
8792
8793
8794
8795
8796
8797
8798
8799
8800
8801
8802
8803
8804

```

```

:*****
:*TEST 52 CHECK CACHE IS BYPASSED ON ASRB OPERAND
;THIS TEST CHECKS THAT THE CACHE IS BYPASSED ON THE
;OPERAND OF THE ASRB INSTRUCTION AND ALSO THE OPERAND
;IS INVALIDATED. TEST-CODE (INCLUDING THE OPERAND
;OF THE ASRB) IS MADE HIT IN GROUP 1. THEN
;ASRB INSTRUCTION IS EXECUTED ON THE CACHED
;OPERAND. IT IS CHECKED IF THE REFERENCE TO THE
;BYTE-OPERAND WAS A MISS. THEN THE SAME OPERAND
;REFERENCED USING AN ORDINARY (NON-BYPASSING)
;INSTRUCTED. AGAIN, THE REFERENCE IS CHECKED FOR
;A MISS.
;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
:*****
TST52: SCOPE

```

```

8798 047074 000004
8799
8800 047076 012703 047076 ASRBCB: MOV #ASRBCB,R3
8801 047102 012704 001000 MOV #1000,R4 ;MAKE TEST-CODE HIT IN GROUP
8802 047106 012737 000034 177746 1$: MOV #SOMOM1,@#CONTRL;1
8803 047114 005763 002000 TST 2000(R3)
8804 047120 012737 000054 177746 MOV #S1MOM1,@#CONTRL

```

```

8805 047126 005723          TST      (R3)+
8806 047130 077412          SOB      R4,1$
8807 047132 042737 000014 177746  BIC      #MOM1,@#CONTRL
8808                                     :EXECUTE AN ASRB AND REFERENCE
8809 047140 106237 047242          ASRB     @#ASLOC      :THE TEST LOCATION
8810 047144 032737 000010 177752  BIT      #10,@#HITMIS
8811 047152 001412          BEQ      2$
8812
8813 047154 013737 177746 001562  MOV      @#CONTRL,$REG0
8814 047162 012737 000001 001564  MOV      #1,$REG1      :GROUP NO.
8815 047170 012737 047242 001566  MOV      #ASLOC,$REG2  :TEST DATA ADDRESS
8816 047176 104076          ERROR    76           :PREVIOUSLY CACHED TEST-LOCATION
8817                                     :WHEN REFERENCED USING AN
8818                                     :ASRB INSTRUCTION WAS NOT
8819                                     :A MISS.
8820                                     :PROBABLE FAILURE = ASRB DOES
8821                                     :NOT FORCE OPERAND-REFERENCE
8822                                     :TO THE MAIN MEMORY
8823
8824 047200 005737 047242          2$: TST      @#ASLOC      :REFERENCE THE TEST-LOCATION
8825 047204 032737 000010 177752  BIT      #10,@#HITMIS  :MISS?
8826 047212 001414          BEQ      TST53        :EXIT
8827
8828 047214 013737 177746 001562  MOV      @#CONTRL,$REG0
8829 047222 012737 000001 001564  MOV      #1,$REG1      :GROUP NO.
8830 047230 012737 047242 001566  MOV      #ASLOC,$REG2  :TEST DATA ADDRESS
8831 047236 104076          ERROR    77           :BYTE-OPERAND (OF ASRB) WAS
8832                                     :NOT INVALIDATED WHEN ASRB
8833                                     :WAS EXECUTED ON A CACHED
8834                                     :LOCATION
8835 047240 000401          BR       TST53        :EXIT
8836
8837 047242 000000          ASLOC: .WORD 0
8838
8839
8840
8841
8842
8843
8844
8845
8846
8847 047244 000004          :*****
8848 047246 013700 177744          :*TEST 53 CHECK CACHE VALID STORE PARITY CHECKER
8849 047252 010037 177744          :THIS TEST FORCES VALID STORE PARITY ERROR IN THE FOUR
8850 047256 005002          :VALID STORES AND CHECKS THE PARITY CHECKERS.
8851 047260 012704 000034          :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8852 047264 012705 000054          :*****
8853 047270 050204          TST53: SCOPE
8854 047272 050205          MOV      @#MSER,R0
8855 047274 010237 177746          MOV      R0,@#MSER
8856 047300 032737 010000 177746  CVSPE: CLR      R2           :BIT MASK FOR VSIU
8857 047306 001374          MOV      #SOMOM1,R4      :SET UP BIT MASKS TO CHECK
8858                                     MOV      #S1MOM1,R5      :GROUP 1 FIRST
8859 047310 010437 177746          CVSPEA: BIS     R2,R4
8860 047314 005737 051370          BIS     R2,R5
          MOV      R2,@#CONTRL
          1$: BIT    #VCIP,@#CONTRL
          BNE    1$
          MOV      R4,@#CONTRL
          TST    @#2$+2000

```

8861											
8862	047320	010537	177746			MOV	R5,@#CONTRL			:MAKE 'NOP' HIT IN GROUP BEING	
8863	047324	005737	047370			TST	@#2\$			:TESTED	
8864											
8865	047	032704	000020			BIT	#S0,R4			:TESTING GROUP 1?	
8866	047334	001004				BNE	13\$			:YES	
8867	047336	042737	000004	177746		BIC	#M0,@#CONTRL			:TESTING GROUP 0,FORCE MISS GROUP 1	
8868	047344	000403				BR	14\$				
8869	047346	042737	000010	177746	13\$:	BIC	#M1,@#CONTRL			:TESTING GROUP 1,FORCE MISS GROUP 0	
8870											
8871	047354	012737	047412	000114	14\$:	MOV	#3\$,@#114			:SETUP PARITY ERROR TRAP VECTOR	
8872											
8873	047362	052737	002000	177746		BIS	#FVPE,@#CONTRL			:FORCE VALID STORE PARITY ERROR	
8874											
8875	047370	000240			2\$:	NOP				:REFERENCE OF THIS INSTRUCTION	
8876										:WILL FORCE A VALID STORE	
8877										:PARITY ERROR. TRAP TO 114	
8878										:SHOULD OCCUR.	
8879	047372	042737	002000	177746		BIC	#FVPE,@#CONTRL			:USER FVPE IF STILL SET	
8880	047400	013737	177746	001562		MOV	@#CONTRL,\$REGO				
8881	047406	104103				ERROR	103			:VALID STORE PARITY ERROR WAS	
8882										:FORCED BY SETTINGS FVPE IN	
8883										:CCR AND MAKING A REFERENCE	
8884										:TO A LOC THAT WAS MADE	
8885										:A HIT. PARITY ERROR TRAP	
8886										:DID NOT OCCUR ON DETECTING	
8887										:THAT PARITY ERROR IN VALID	
8888	047410	000411				BR	6\$			:STORE	
8889											
8890	047412				3\$:					:ENTER HERE IF A PARITY ERROR	
8891										:TRAP OCCURS AS EXPECTED	
8892	047412	022626				CMP	(SP)+,(SP)+			:POP THE STACK	
8893	047414	032737	002000	177746	5\$:	BIT	#FVPE,@#CONTRL			:FVPE CLEARED AFTER PARITY ERROR?	
8894	047422	001404				BEQ	6\$				
8895	047424	013737	177746	001562		MOV	@#CONTRL,\$REGO				
8896	047432	104105				ERROR	105			:FVPE DID NOT GET CLEARED	
8897										:AFTER VALID SOTRE PARITY ERROR	
8898										:OCCURED	
8899											
8900	047434	032737	100000	177746	6\$:	BIT	#VSPE,@#CONTRL			:DID-VALID-STORE-PARITY-ERROR SET?	
8901	047442	001004				BNE	7\$				
8902	047444	013737	177746	001562		MOV	@#CONTRL,\$REGO				
8903	047452	104106				ERROR	106			:VALID-STORE-PARITY-ERROR BIT	
8904										:DID NOT SET IN CCR WHEN	
8905										:PARITY ERROR FROM V-STORE)	
8906	047454	005003			7\$:	CLR	R3			:WAS FORCED	
8907										:TESTING GROUP 1?	
8908	047456	032705	000040			BIT	#S1,R5			:YES	
8909	047462	001003				BNE	8\$				
8910	047464	012700	000020			MOV	#BIT4,R0			:SET BIT MASK FOR GROUP 0	
8911	047470	000402				BR	9\$				
8912	047472	012700	000040		8\$:	MOV	#BIT5,R0			:SET BIT MASK FOR GROUP 1	
8913	047476	013701	177744		9\$:	MOV	@#MSER,R1			:GROUP IN WHICH THE PARITY ERROR	
8914	047502	042701	177716			BIC	#^CBIT4+^CBIT5,R1			:OCCURED) WAS SET IN MEMORY	
8915	047506	020100				CMP	R1,R0			:SYSTEM ERROR REGISTER	
8916	047510	001413				BEQ	10\$				

```

8917 047512 013737 177746 001562      MOV    @#CONTRL,$REG0
8918 047520 013737 177744 001564      MOV    @#MSER,$REG1
8919 047526 010337 001566              MOV    R3,$REG2      ;GROUP NO. BEING TESTED
8920 047532 010037 001570              MOV    R0,$REG3      ;EXPECTED BITS (4,5) IN MEM SY
8921                                     ;ERROR REGISTER
8922 047536 104107                      ERROR  107           ;PROPER BITS (ADDRESS MEMORY)
8923                                     ;PARITY ERROR.(4,5) WERE NOT
8924                                     ;SET IN MEM-SYS.ERROR
8925                                     ;REGISTER WHEN VALID STORE
8926                                     ;PARITY ERROR WAS FORCED.
8927
8928 047540 013703 177746      10$:  MOV    @#CONTRL,R3      ;DID VSIU BIT SWITCH AFTER
8929 047544 074203              XOR    R2,R3          ;PARITY ERROR TRAP?
8930 047546 032703 020000      BIT    #VSIU,R3
8931 047552 001404              BEQ    11$            ;NO, OK
8932 047554 013737 177746 001562      MOV    @#CONTRL,$REG0
8933 047562 104110              ERROR  110           ;VSIU SWITCHED WHEN A VALID
8934                                     ;STORE PARITY ERROR WAS
8935                                     ;FORCED. IT SHOULD NOT.
8936 047564 013701 177744      11$:  MOV    @#MSER,R1      ;CLEAR MEMORY SYSTEM ERROR
8937 047570 010137 177744      MOV    R1,@#MSER     ;REGISTER
8938 047574 005737 177744      TST    @#MSER         ;CLEARED?
8939 047600 001404              BEQ    12$
8940 047602 013737 177744 001562      MOV    @#MSER,$REG0
8941 047610 104111              ERROR  111           ;MEMORY SYSTEM ERROR REGISTER
8942                                     ;WOULD NOT BE CLEARED BY
8943                                     ;WRITING ITS CONTENTS BACK
8944                                     ;INTO ITSELF. NOTE PREVIOUSLY
8945                                     ;A VALID STORE PARITY ERROR
8946                                     ;OCCURED THIS SETTINGS ?
8947                                     ;BIT 4 OR 5 IN IT.
8948 047612 012700 020000      12$:  MOV    #VSIU,R0
8949 047616 074002              XOR    R0,R2          ;DONE VALID STORE B?
8950 047620 001223              BNE    CVSPEA        ;GO CHECK VALID STORE B
8951
8952 047622 032705 000020      BIT    #S0,R5        ;CHECKED GROUP 0?
8953 047626 001007              BNE    TST54         ;:EXIT
8954
8955 047630 012704 000054      MOV    #S1MOM1,R4    ;SET UP BIT MASKS TO CHECK
8956 047634 012705 000034      MOV    #S0MOM1,R5    ;GROUP 0
8957 047640 005002              CLR    R2             ;BUT MASK FOR VALID STORE
8958 047642 000137 047270      JMP    CVSPEA
8959
8960      ;*****
8961      ;*TEST 54      CHECK THAT CACHE-MISS OCCURS ON A VALID STORE PARITY ERROR
8962      ;THIS TEST FORCES A VALID STORE PARITY ERROR AND CHECKS
8963      ;THAT A MISS OCCURS ON THE REFERENCE THAT CAUSED
8964      ;THE PARITY ERROR. THE CACHE LOCATION THAT GAVE THE
8965      ;PARITY ERROR IS INVALIDATED AND A SLOW CYCLE IS
8966      ;PERFORMED TO THE MAIN MEMORY. THIS TEST IS
8967      ;PERFORMED WITH THE 'DISABLE TRAPS' BIT OF THE
8968      ;CACHE CONTROL REGISTER SET, THUS A PARITY ERROR
8969      ;TRAP WILL NOT OCCUR. THIS IS DONE SO THAT THE
8970      ;HIT-MISS REGISTER CAN BE READ WITHOUT LOSING
8971      ;THE INFORMATION CONTAINED IN IT.
8972      ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
    
```

```

*****
*TEST 54      CHECK THAT CACHE-MISS OCCURS ON A VALID STORE PARITY ERROR
;THIS TEST FORCES A VALID STORE PARITY ERROR AND CHECKS
;THAT A MISS OCCURS ON THE REFERENCE THAT CAUSED
;THE PARITY ERROR. THE CACHE LOCATION THAT GAVE THE
;PARITY ERROR IS INVALIDATED AND A SLOW CYCLE IS
;PERFORMED TO THE MAIN MEMORY. THIS TEST IS
;PERFORMED WITH THE 'DISABLE TRAPS' BIT OF THE
;CACHE CONTROL REGISTER SET, THUS A PARITY ERROR
;TRAP WILL NOT OCCUR. THIS IS DONE SO THAT THE
;HIT-MISS REGISTER CAN BE READ WITHOUT LOSING
;THE INFORMATION CONTAINED IN IT.
;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
*****
    
```

```

8973 047646 000004          TST54: SCOPE
8974 047650 005002          VSCM: CLR      R2          ;BIT MASK FOR VSIU
8975                                     ;TRAPS
8976 047652 012704 000034          MOV      #SOMOM1,R4      ;SET BIT MASKS TO CHECK
8977 047656 012705 000054          MOV      #S1MOM1,R5      ;GROUP 1 FIRST
8978 047662 050204          VSCMA: BIS      R2,R4
8979 047664 050205          BIS      R2,R5
8980 047666 010237 177746          MOV      R2,@#CONTRL
8981 047672 032737 010000 177746 1$: BIT      #VCIP,@#CONTRL
8982 047700 001374          BNE      1$
8983
8984 047702 010437 177746          MOV      R4,@#CONTRL      ;MAKE 'NOP' LIST IN GROUP
8985 047706 005737 051736          TST      @#2$+2000      ;BEING TESTED
8986 047712 010537 177746          MOV      R5,@#CONTRL
8987 047716 005737 047736          TST      @#2$
8988 047722 042737 000014 177746          BIC      #MOM1,@#CONTRL
8989 047730 052737 002001 177746          BIS      #FVPE+DT,@#CONTRL ;FORCE VALID STORE PARITY ERROR
8990
8991 047736 000240          2$: NOP      ;REFERENCE OF THIS INSTRUCTION
8992                                     ;WILL FORCE A VALID STORE
8993                                     ;PARITY ERROR
8994
8995 047740 032737 000010 177752          BIT      #10,@#HITMIS    ;CHECK THAT THE REFERENCE
8996 047746 001407          BEQ      3$              ;WHICH CAUSED THE V-STORE
8997 047750 013737 177746 001562          MOV      @#CONTRL,$REGO  ;PARITY ERROR WAS A MISS
8998 047756 013737 177744 001562          MOV      @#MSER,$REG1   ;TEXT-DATA REFERENCE WHICH
8999 047764 104104          ERROR    104            ;CAUSED A PARITY ERROR
9000                                     ;(IN THE VALID STORE)
9001                                     ;SHOULD HAVE BEEN A MISS-
9002                                     ;IT WAS NOT.
9003
9004 047766 032737 100000 177746          3$: BIT      #VSPE,@#CONTRL ;DID VALID STORE PARITY ERROR
9005                                     ;SET?
9006 047774 001004          BNE      4$
9007 047776 013737 177746 001562          MOV      @#CONTRL,$REGO
9008 050004 104106          ERROR    106            ;VALID STORE PARITY ERROR BIT
9009                                     ;DID NOT SET IN CCR WHEN
9010                                     ;PARITY ERROR FROM (V-STORE)
9011                                     ;WAS FORCED
9012
9013 050006 052737 100000 177746          4$: BIS      #VSPE,@#CONTRL ;CLEAR VSPE
9014 050014 032737 100000 177746          BIT      #VSPE,@#CONTRL ;CHECK
9015 050022 001404          BEQ      5$
9016 050024 013737 177746 001562          MOV      @#CONTRL,$REGO
9017 050032 104112          ERROR    112            ;VALID STORE PARITY ERROR
9018                                     ;BIT COULD NOT BE CLEARED IN CCR
9019
9020 050034 013737 177744 177744          5$: MOV      @#MSER,@#MSER ;CLEAR MEMORY SYSTEM ERROR
9021                                     ;REGISTER
9022 050042 012700 020000          MOV      #VSIU,R0
9023 050046 074002          XOR      R0,R2          ;DONE VALID STORE B?
9024 050050 001304          BNE      VSCMA          ;GO CHECK V-STORE B
9025 050052 032705 000020          BIT      #S0,R5        ;CHECKED GROUP 0
9026 050056 001007          BNE      6$
9027
9028 050060 012704 000054          MOV      #S1MOM1,R4      ;SET UP BIT MASKS ITS CHECK

```

```

9029 050064 012705 000034      MOV      #SOMOM1,R5      ;GROUP 0
9030 050070 005002              CLR      R2
9031 050072 000137 047662      JMP      VSCMA
9032
9033 050076 005037 177746      6$: CLR      @#CONTRL
9034 050102                      KT:
9035
9036
9037
9038
9039
9040
9041 050102 000004              ;*****
9042 050104 012737 000012 001702  ;*TEST 55      CHECK BYP ON KERNEL PAGE BITS
9043 050112 105737 001750              ;THIS TEST IS EXECUTED ONLY ON KB11-E,KB11-EM,AND MODIFIED KB11-B/C (KB11CM)
9044 050116 001003              ;*****
9045 050120 005737 001746      TST55: SCOPE
9046 050124 001444              MOV      #12,$TIMES      ;;DO 12 ITERATIONS
9047 050126 012700 172300      TSTB     KB11CM
9048 050132 005010 4$: CLR      (R0)          ;BR IF MOIFIED 11/70 (KB11CM)
9049 050134 032710 100000      BIT      #BYP,(R0)       ;IS IT A KB11-E OR KB11-EM?
9050 050140 001405              BEQ      TST56           ;;
9051 050142 010037 001562      5$: MOV      #KIPDR0,R0   ;POINT TO KIPDR0
9052 050146 011037 001564      4$: CLR      (R0)          ;CLEAR KIPDR
9053 050152 104123              BEQ      1$             ;DID BYP CLEAR?
9054
9055 050154 052710 100000      1$: BIS      #BYP,(R0)    ;BRANCH IF YES
9056 050160 032710 100000      BIT      #BYP,(R0)       ;SET BYP
9057 050164 001005              BNE      2$             ;IS IT SET?
9058 050166 010037 001562      MOV      R0,$REG0       ;BRANCH IF YES
9059 050172 011037 001564      MOV      (R0),$REG1
9060 050176 104124              ERROR    123           ;BYP STUCK SET
9061
9062 050200 042710 100000      1$: BIS      #BYP,(R0)    ;SET BYP
9063 050204 032710 100000      BIT      #BYP,(R0)       ;IS IT SET?
9064 050210 001405              BNE      2$             ;BRANCH IF YES
9065 050212 010037 001562      MOV      R0,$REG0
9066 050216 011037 001564      MOV      (R0),$REG1
9067 050222 104123              ERROR    123           ;BYP STUCK SET
9068
9069 050224 062700 000002      3$: ADD      #2,R0        ;POINT TO NEXT PDR
9070 050230 020027 172340      CMP      R0,#KDPDR7+2   ;ARE WE FINISHED?
9071 050234 001336              BNE      4$             ;BRANCH IF NOT
9072
9073
9074
9075
9076
9077 050236 000004              ;*****
9078 050240 012737 000012 001702  ;*TEST 56      CHECK BYP ON SUPERVISOR PAGE BITS
9079 050246 105737 001750              ;THIS TEST IS EXECUTED ONLY ON KB11-E, KB11-EM, AND MODIFIED KB11-B/C (KB11CM).
9080 050252 001003              ;*****
9081 050254 005737 001746      TST56: SCOPE
9082 050260 001444              MOV      #12,$TIMES      ;;DO 12 ITERATIONS
9083 050262 012700 172200      TSTB     KB11CM
9084 050266 005010 4$: CLR      (R0)          ;BR IF MOIFIED 11/70 (KB11CM)

```



```

9085 050270 032710 100000 BIT #BYP,(R0) ;DID BYP CLEAR?
9086 050274 001405 BEQ 1$ ;BRANCH IF YES
9087 050276 010037 001562 MOV R0,$REG0
9088 050302 011037 001564 MOV (R0),$REG1
9089 050306 104130 ERROR 130 ;BYP STUCK SET
9090
9091 050310 052710 100000 1$: BIS #BYP,(R0) ;SET BYP
9092 050314 032710 100000 BIT #BYP,(R0) ;DID IT SET?
9093 050320 001005 BNE 2$ ;BRANCH IF YES
9094 050322 010037 001562 MOV R0,$REG0
9095 050326 011037 001564 MOV (R0),$REG1
9096 050332 104131 ERROR 131 ;BYP STUCK CLEAR
9097
9098 050334 042710 100000 2$: BIC #BYP,(R0) ;CLEAR BYP
9099 050340 032710 100000 BIT #BYP,(R0) ;IS IT CLEAR?
9100 050344 001405 BEQ 3$ ;BRANCH IF YES
9101 050346 010037 001562 MOV R0,$REG0
9102 050352 011037 001564 MOV (R0),$REG1
9103 050356 104130 ERROR 130 ;BYP STUCK SET
9104
9105 050360 062700 000002 3$: ADD #2,R0 ;POINT TO NEXT PDR
9106 050364 020027 172240 CMP R0,#SDPDR7+2 ;ARE WE FINISHED?
9107 050370 001336 BNE 4$ ;BRANCH IF NO
9108
9109 ;*****
9110 ;*TEST 57 CHECK BYP ON USER PAGE BITS
9111 ;*THIS TEST IS EXECUTED ONLY ON KB11-E, KB11-EM, AND MODIFIED KB11-B/C (KB11CM).
9112 ;*****
9112 050372 000004 TST57: SCOPE
9113 050374 012737 000012 001702 MOV #12,$TIMES ;DO 12 ITERATIONS
9114 050402 105737 001750 TSTB KB11CM
9115 050406 001003 BNE 5$ ;BR IF MOIFIED 11/70 (KB11CM)
9116 050410 005737 001746 TST KB11E ;IS IT A KB11-E OR KB11-EM?
9117 050414 001444 BEQ TST60 ;
9118 050416 012700 177600 5$: MOV #UIPDRO,R0 ;POINT TO UIPDR0
9119 050422 005010 4$: CLR (R0) ;CLEAR UIPDR
9120 050424 032710 100000 BIT #BYP,(R0) ;DID BYP CLEAR?
9121 050430 001405 BEQ 1$ ;BRANCH IF YES
9122 050432 010037 001562 MOV R0,$REG0
9123 050436 011037 001564 MOV (R0),$REG1
9124 050442 104132 ERROR 132 ;BYP STUCK SET
9125
9126 050444 052710 100000 1$: BIS #BYP,(R0) ;SET BYP
9127 050450 032710 100000 BIT #BYP,(R0) ;IS IT SET?
9128 050454 001005 BNE 2$ ;BRANCH IF YES
9129 050456 010037 001562 MOV R0,$REG0
9130 050462 011037 001564 MOV (R0),$REG1
9131 050466 104133 ERROR 133 ;BYP STUCK CLEAR
9132
9133 050470 042710 100000 2$: BIC #BYP,(R0) ;CLEAR BYP
9134 050474 032710 100000 BIT #BYP,(R0) ;IS IT CLEAR?
9135 050500 001405 BEQ 3$ ;BRANCH IF YES
9136 050502 010037 001562 MOV R0,$REG0
9137 050506 011037 001564 MOV (R0),$REG1
9138 050512 104132 ERROR 132 ;BYP STUCK SET
9139
9140 050514 062700 000002 3$: ADD #2,R0 ;POINT TO NEXT PDR

```





```

9253                                     ;(BEING TESTED). BYPASS BIT WAS
9254                                     ;SET IN THE PDR AND TEST-LOC
9255                                     ;WAS REFERENCED. IT SHOULD HAVE
9256                                     ;BEEN A MISS (BECAUSE OF BYPASS)
9257                                     ;PROBABLE FAULT: SETTING CACHE
9258                                     ;BYPASS IN PDR DOES NOT BYPASS
9259                                     ;VIRTUAL REFERENCES MAPPED THRU
9260                                     ;THAT PAGE.
9261 051214 012737 000001 177572      MOV    #1,@MMR0      ;TURN MM BACK ON
9262
9263
9264 051222 062700 000002      6$:    ADD    #2,R0
9265 051226 077131              SOB    R1,5$
9266 051230 042713 100000      BIC    #BYP,(R3)    ;CLEAR BYPASS IN PDR
9267 051234 012701 001000      MOV    #1000,R1    ;REFERENCE THE TEST-DATA AND
9268                                     ;MAKE SURE IT WAS INVALIDATED
9269                                     ;ON PREVIOUS BYPASS
9270 051240 005712              7$:    TST    (R2)    ;REFERENCE TEST DATA
9271 051242 032737 000010 177746      BIT    #10,@CONTRL ;MISS?
9272 051250 001421              BEQ    8$
9273 051252 013737 177746 001562      MOV    @CONTRL,$REG0
9274 051260 010537 001564      MOV    R5,$REG1    ;PAR ADDRESS
9275 051264 011537 001566      MOV    (R5),$REG2  ;PAR CONTENTS
9276 051270 011337 001570      MOV    (R3),$REG3  ;PDR CONTENTS
9277 051274 010237 001572      MOV    R2,$REG4    ;TEST DATA ADDRESS (VIRTUAL ADDRESS)
9278 051300 005037 177572      CLR    @MMR0      ;TURN OFF MM
9279 051304 104127              ERROR 127        ;TEST-DATA REFERENCE WAS NOT
9280                                     ;A MISS. PROBABLE FAILURE:
9281                                     ;PREVIOUSLY CACHED TEST DATA
9282                                     ;LOCATIONS WERE NOT INVALIDATED
9283                                     ;(IN THE CACHE) WHEN CACHE
9284                                     ;BYPASS WAS FORCED ON REFERENCES THROUGH
9285                                     ;THE VIRTUAL PAGE (BEING TESTED).
9286 051306 012737 000001 177572      MOV    #1,@MMR0      ;TURN MM BACK ON
9287
9288 051314 062702 000002      8$:    ADD    #2,R2
9289 051320 077131              SOB    R1,7$
9290
9291
9292 051322 005037 177572      CLR    @MMR0      ;DISABLE KT
9293 051326 005037 172516      CLR    @MMR3
9294 051332 062704 020000      ADD    #20000,R4   ;INITIALIZE APF FIELD MASK FOR THE
9295 051336 062705 000002      ADD    #2,R5      ;NEXT PAR TO BE TESTED
9296 051342 062703 000002      ADD    #2,R3      ;NEXT PDR TO BE TESTED
9297
9298 051346 020327 172316      CMP    R3,#KIPDR0+16 ;DONE TESTING EVERY PDR?
9299 051352 001402              BEQ    VPBPE
9300 051354 000137 050602      JMP    VPBPA
9301 051360
9302 VPBPE:
9303 .SBTTL END OF PASS ROUTINE
9304
9305 ;*****
9306 ;*INCREMENT THE PASS NUMBER ($PASS)
9307 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
9308 ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
9309 ;*IF THERES A MONITOR GO TO IT

```

```

9309          ;*IF THERE ISN'T JUMP TO LOOP
9310
9311 051360    $EOP:
9312 051360 000004    SCOPE
9313 051362 005037 001502    CLR    $STSTNM    ;;ZERO THE TEST NUMBER
9314 051366 005037 001702    CLR    $TIMES    ;;ZERO THE NUMBER OF ITERATIONS
9315 051372 005237 001724    INC    $PASS    ;;INCREMENT THE PASS NUMBER
9316 051376 042737 100000 001724    BIC    #100000,$PASS    ;;DON'T ALLOW A NEG. NUMBER
9317 051404 005327    DEC    (PC)+    ;;LOOP?
9318 051406 000001    $EOPCT: .WORD    1
9319 051410 003031    BGT    $DOAGN    ;;YES
9320 051412 012737    MOV    (PC)+,@(PC)+    ;;RESTORE COUNTER
9321 051414 000001    $ENDCT: .WORD    1
9322 051416 051406    $EOPCT
9323 051420 104401 051503    TYPE    $SENDMG    ;;TYPE 'END PASS #'
9324 051424 013746 001724    MOV    $PASS,-(SP)    ;;SAVE $PASS FOR TYPEOUT
9325 051430 104405    TYPDS    ;;GO TYPE--DECIMAL ASCII WITH SIGN
9326 051432 104401 051500    TYPE    $NULL    ;;TYPE A NULL CHARACTER
9327 051436 013700 000042    $GET42: MOV    @#42,R0    ;;GET MONITOR ADDRESS
9328 051442 001414    BEQ    $DOAGN    ;;BRANCH IF NO MONITOR
9329 051444 012703 125252    MOV    #125252,R3
9330 051450 004737 056106    JSR    PC,CHAINQ
9331 051454 013700 000042    MOV    @#42,R0    ;;INSURE R0 CONTAINS THE MONITORS
9332 051460 001405    BEQ    $DOAGN    ;;RETURN ADDRESS
9333 051462 000005    RESET    ;;CLEAR THE WORLD
9334 051464 004710    $ENDAD: JSR    PC,(R0)    ;;GO TO MONITOR
9335 051466 000240    NOP    ;;SAVE ROOM
9336 051470 000240    NOP    ;;FOR
9337 051472 000240    NOP    ;;ACT11
9338 051474    $DOAGN:
9339 051474 000137    JMP    @(PC)+    ;;RETURN
9340 051476 005306    $RTNAD: .WORD    LOOP
9341 051500    377    377    000    $NULL: .BYTE    -1,-1,0    ;;NULL CHARACTER STRING
9342 051503    015    042412 042116    $ENDMG: .ASCIIZ <15><12>/END PASS #/
9343 051510 050040 051501 020123
9344 051516 000043
9345
9346          .SBTTL SCOPE HANDLER ROUTINE
9347
9348          ;*****
9349          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
9350          ;*AND LOAD THE TEST NUMBER($STSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
9351          ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
9352          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
9353          ;*SW14=1    LOOP ON TEST
9354          ;*SW11=1    INHIBIT ITERATIONS
9355          ;*SW09=1    LOOP ON ERROR
9356          ;*SW08=1    LOOP ON TEST IN SWR<6:0>
9357          ;*CALL
9358          ;*    SCOPE    ;;SCOPE=IOT
9359
9360          $SCOPE:
9361 051520 104407    CKSWR    ;;TEST FOR CHANGE IN SOFT-SWR
9362 051522 032777 040000 130010 1$: BIT    #BIT14,@SWR    ;;LOOP ON PRESENT TEST?
9363 051530 001117    BNE    $OVER    ;;YES IF SW14=1
9364          ;*****START OF CODE FOR THE XOR TESTER*****
    
```

```

9365 051532 000416          $XTSTR: BR      6$          ::IF RUNNING ON THE 'XOR' TESTER CHANGE
9366                                ::THIS INSTRUCTION TO A 'NOP' (NOP=240)
9367 051534 013746 000004          MOV      @WERRVEC,-(SP)      ::SAVE THE CONTENTS OF THE ERROR VECTOR
9368 051540 012737 051560 000004          MOV      #5$,@WERRVEC      ::SET FOR TIMEOUT
9369 051546 005737 177060          TST      @#177060          ::TIME OUT ON XOR?
9370 051552 012637 000004          MOV      (SP)+,@WERRVEC    ::RESTORE THE ERROR VECTOR
9371 051556 000466          BR      $SVLAD             ::GO TO THE NEXT TEST
9372 051560 022626          5$:    CMP      (SP)+,(SP)+      ::CLEAR THE STACK AFTER A TIME OUT
9373 051562 012637 000004          MOV      (SP)+,@WERRVEC    ::RESTORE THE ERROR VECTOR
9374 051566 000426          BR      7$                ::LOOP ON THE PRESENT TEST
9375 051570          6$:;#####END OF CODE FOR THE XOR TESTER#####
9376 051570 032777 000400 127742          BIT      #BIT08,@SWR       ::LOOP ON SPEC. TEST?
9377 051576 001407          BEQ      2$                ::BR IF NO
9378 051600 017746 127734          MOV      @SWR,-(SP)        ::SET DESIRED TEST NUM. FROM SWR
9379 051604 042716 000200          BIC      #SSWRMK,(SP)      ::STRIP AWAY UNDESIRED BITS
9380 051610 122637 001502          CMPB     (SP)+,$TSTNM      ::ON THE RIGHT TEST?
9381 051614 001465          BEQ      $OVER            ::BR IF YES
9382 051616 105737 001503          2$:    TSTB     $ERFLG       ::HAS AN ERROR OCCURRED?
9383 051622 001421          BEQ      3$                ::BR IF NO
9384 051624 123737 001515 001503          CMPB     $ERMAX,$ERFLG    ::MAX. ERRORS FOR THIS TEST OCCURRED?
9385 051632 101015          BHI      3$                ::BR IF NO
9386 051634 032777 001000 127676          BIT      #BIT09,@SWR       ::LOOP ON ERROR?
9387 051642 001404          BEQ      4$                ::BR IF NO
9388 051644 013737 001510 001506          7$:    MOV      $LPERR,$LPADR  ::SET LOOP ADDRESS TO LAST SCOPE
9389 051652 000446          BR      $OVER
9390 051654 105037 001503          4$:    CLRB     $ERFLG       ::ZERO THE ERROR FLAG
9391 051660 005037 001702          CLR      $TIMES           ::CLEAR THE NUMBER OF ITERATIONS TO MAKE
9392 051664 000415          BR      1$                ::ESCAPE TO THE NEXT TEST
9393 051666 032777 004000 127644          3$:    BIT      #BIT11,@SWR    ::INHIBIT ITERATIONS?
9394 051674 001011          BNE      1$                ::BR IF YES
9395 051676 005737 001724          TST      $PASS            ::IF FIRST PASS OF PROGRAM
9396 051702 001406          BEQ      1$                ::INHIBIT ITERATIONS
9397 051704 005237 001504          INC      $ICNT            ::INCREMENT ITERATION COUNT
9398 051710 023737 001702 001504          CMP      $TIMES,$ICNT     ::CHECK THE NUMBER OF ITERATIONS MADE
9399 051716 002024          BGE      $OVER            ::BR IF MORE ITERATION REQUIRED
9400 051720 012737 000001 001504          1$:    MOV      #1,$ICNT      ::REINITIALIZE THE ITERATION COUNTER
9401 051726 013737 052004 001702          MOV      $MXCNT,$TIMES    ::SET NUMBER OF ITERATIONS TO DO
9402 051734 105237 001502          $SVLAD: INCB     $TSTNM     ::COUNT TEST NUMBERS
9403 051740 113737 001502 001722          MOVB    $TSTNM,$TESTNM    ::SET TEST NUMBER IN APT MAILBOX
9404 051746 011637 001506          MOV      (SP),$LPADR      ::SAVE SCOPE LOOP ADDRESS
9405 051752 011637 001510          MOV      (SP),$LPERR      ::SAVE ERROR LOOP ADDRESS
9406 051756 005037 001704          CLR      $ESCAPE         ::CLEAR THE ESCAPE FROM ERROR ADDRESS
9407 051762 112737 000001 001515          MOVB    #1,$ERMAX        ::ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9408 051770 013777 001502 127544          $OVER: MOV      $TSTNM,@DISPLAY ::DISPLAY TEST NUMBER
9409 051776 013716 001506          MOV      $LPADR,(SP)     ::FUDGE RETURN ADDRESS
9410 052002 000002          RTI                      ::FIXES PS
9411 052004 000001          $MXCNT: 1                ::MAX. NUMBER OF ITERATIONS
    
```

```

9412
9413          .SBTTL  ERROR HANDLER ROUTINE
9414
9415          ::*****
9416          ::*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
9417          ::*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
9418          ::*AND GO TO ERTYPE ON ERROR
9419          ::*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
9420          ::*SW15=1          HALT ON ERROR
    
```

```
9421          ;*SW13=1          INHIBIT ERROR TYPEOUTS
9422          ;*SW10=1          BELL ON ERROR
9423          ;*SW09=1         LOOP ON ERROR
9424          ;*CALL
9425          ;*          ERROR N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER
9426
9427          $ERROR:
9428          052006 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
9429          052010 105237 001503 7$:          INCB          $ERFLG          ;;SET THE ERROR FLAG
9430          052014 001775          BEQ          7$          ;;DON'T LET THE FLAG GO TO ZERO
9431          052016 013777 001502 127516          MOV          $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
9432          052024 032777 002000 127506          BIT          #BIT10,@SWR   ;;BELL ON ERROR?
9433          052032 001402          BEQ          1$          ;;NO - SKIP
9434          052034 104401 001706          TYPE          $BELL          ;;RING BELL
9435          052040 005237 001512          1$:          INC          $ERTTL          ;;COUNT THE NUMBER OF ERRORS
9436          052044 011637 001516          MOV          (SP),$ERRPC   ;;GET ADDRESS OF ERROR INSTRUCTION
9437          052050 162737 000002 001516          SUB          #2,$ERRPC
9438          052056 117737 127434 001514          MOVB         @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
9439          052064 032777 020000 127446          BIT          #BIT13,@SWR   ;;SKIP TYPEOUT IF SET
9440          052072 001004          BNE          20$          ;;SKIP TYPEOUTS
9441          052074 004737 056354          JSR          PC,ERTYPE    ;;GO TO USER ERROR ROUTINE
9442          052100 104401 001713          TYPE          $CRLF
9443          052104          20$:
9444          052104 122737 000001 001736          CMPB         #APTENV,$ENV  ;;RUNNING IN APT MODE
9445          052112 001007          BNE          2$          ;;NO,SKIP APT ERROR REPORT
9446          052114 113737 001514 052126          MOVB         $ITEMB,21$   ;;SET ITEM NUMBER AS ERROR NUMBER
9447          052122 004737 052236          JSR          PC,$ATY4     ;;REPORT FATAL ERROR TO APT
9448          052126          000
9449          052127          000          21$:          .BYTE          0
9450          052130 000777          22$:          BR          22$          ;;APT ERROR LOOP
9451          052132 005777 127402          2$:          TST          @SWR          ;;HALT ON ERROR
9452          052136 100002          BPL          3$          ;;SKIP IF CONTINUE
9453          052140 000000          HALT          ;;HALT ON ERROR!
9454          052142 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
9455          052144 032777 001000 127366          3$:          BIT          #BIT09,@SWR  ;;LOOP ON ERROR SWITCH SET?
9456          052152 001402          BEQ          4$          ;;BR IF NO
9457          052154 013716 001510          MOV          $LPERR,(SP)  ;;FUDGE RETURN FOR LOOPING
9458          052160 005737 001704          4$:          TST          $ESCAPE     ;;CHECK FOR AN ESCAPE ADDRESS
9459          052164 001402          BEQ          5$          ;;BR IF NONE
9460          052166 013716 001704          MOV          $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
9461          052172          5$:
9462          052172 022737 051464 000042          CMP          #SENDAD,@#42 ;;ACT-11 AUTO-ACCEPT?
9463          052200 001001          BNE          6$          ;;BRANCH IF NO
9464          052202 000000          HALT          ;;YES
9465          052204          6$:
9466          052204 012737 177777 177744          MOV          #-1,@MEMERR
9467          052212 005037 177766          CLR          @CPUERR
9468          052216 000002          RTI
9469
9470
9471          .SBTTL APT COMMUNICATIONS ROUTINE
9472
9473          ;*****
9474          052220 112737 000001 052464 $ATY1: MOVB          #1,$FFLG          ;;TO REPORT FATAL ERROR
9475          052226 112737 000001 052462 $ATY3: MOVB          #1,$MFLG          ;;TO TYPE A MESSAGE
9476          052234 000403          BR          $ATYC
```

```

9477 052236 112737 000001 052464 $ATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
9478 052244 $ATYC:
9479 052244 010046 MOV RO,-(SP) ;;PUSH RO ON STACK
9480 052246 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
9481 052250 105737 052462 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
9482 052254 001450 BEQ 5$ ;;IF NOT: BR
9483 052256 122737 000001 001736 CMPB #APTENV,$ENV ;;OPERATING UNDER APT?
9484 052264 001031 BNE 3$ ;;IF NOT: BR
9485 052266 132737 000100 001737 BITB #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
9486 052274 001425 BEQ 3$ ;;IF NOT: BR
9487 052276 017600 000004 MOV @4(SP),RO ;;GET MESSAGE ADDR.
9488 052302 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
9489 052310 005737 001716 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
9490 052314 001375 BNE 1$ ;;IF NOT: WAIT
9491 052316 010037 001732 MOV RO,$MSGAD ;;PUT ADDR IN MAILBOX
9492 052322 105720 2$: TSTB (RO)+ ;;FIND END OF MESSAGE
9493 052324 001376 BNE 2$
9494 052326 163700 001732 SUB $MSGAD,RO ;;SUB START OF MESSAGE
9495 052332 006200 ASR RO ;;GET MESSAGE LNTH IN WORDS
9496 052334 010037 001734 MOV RO,$MSGGLT ;;PUT LENGTH IN MAILBOX
9497 052340 012737 000004 001716 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
9498 052346 000413 BR 5$
9499 052350 017637 000004 052374 3$: MOV @4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
9500 052356 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
9501 052364 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
9502 052370 004737 053336 JSR PC,$TYPE ;;CALL TYPE MACRO
9503 052374 000000 4$: .WORD 0
9504 052376 5$:
9505 052376 105737 052464 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
9506 052402 001416 BEQ 12$ ;;IF NOT: BR
9507 052404 005737 001736 TST $ENV ;;RUNNING UNDER APT?
9508 052410 001413 BEQ 12$ ;;IF NOT: BR
9509 052412 005737 001716 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
9510 052416 001375 BNE 11$ ;;IF NOT: WAIT
9511 052420 017637 000004 001720 MOV @4(SP),$FATAL ;;GET ERROR #
9512 052426 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
9513 052434 005237 001716 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
9514 052440 105037 052464 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
9515 052444 105037 052463 CLRB $LFLG ;;CLEAR LOG FLAG
9516 052450 105037 052462 CLRB $MFLG ;;CLEAR MESSAGE FLAG
9517 052454 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
9518 052456 012600 MOV (SP)+,RO ;;POP STACK INTO RO
9519 052460 000207 RTS PC ;;RETURN
9520 052462 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
9521 052463 000 $LFLG: .BYTE 0 ;;LOG FLAG
9522 052464 000 $FFLG: .BYTE 0 ;;FATAL FLAG
9523 052466 .EVEN
9524 000200 APTSIZE=200
9525 000001 APTENV=001
9526 000100 APTSPOOL=100
9527 000040 APTCSUP=040
9528 .SBTTL TTY INPUT ROUTINE
9529
9530 *****
9531 .ENABL LSB
9532

```



```

9533
9534
9535
9536
9537
9538 052466 022737 000176 001540 $CKSWR: CMP #SWREG,SWR ::IS THE SOFT-SWR SELECTED?
9539 052474 001074 BNE 15$ ::BRANCH IF NO
9540 052476 105777 127042 TSTB @TKS ::CHAR THERE?
9541 052502 100071 BPL 15$ ::IF NO, DON'T WAIT AROUND
9542 052504 117746 127036 MOVB @TKB,-(SP) ::SAVE THE CHAR
9543 052510 042716 177600 BIC #^C177,(SP) ::STRIP-OFF THE ASCII
9544 052514 022726 000007 CMP #7,(SP)+ ::IS IT A CONTROL G?
9545 052520 001062 BNE 15$ ::NO, RETURN TO USER
9546 052522 123727 001534 000001 CMPB $AUTOB,#1 ::ARE WE RUNNING IN AUTO-MODE?
9547 052530 001456 BEQ 15$ ::BRANCH IF YES
9548
9549 052532 104401 053213 $GTSWR: TYPE ,SCNTLG ::ECHO THE CONTROL-G (^G)
9550 052536 104401 053220 TYPE ,SMSWR ::TYPE CURRENT CONTENTS
9551 052542 013746 000176 MOV SWREG,-(SP) ::SAVE SWREG FOR TYPEOUT
9552 052546 104402 TYPOC ::GO TYPE--OCTAL ASCII(ALL DIGITS)
9553 052550 104401 053231 TYPE ,SMNEW ::PROMPT FOR NEW SWR
9554 052554 005046 19$: CLR -(SP) ::CLEAR COUNTER
9555 052556 005046 CLR -(SP) ::THE NEW SWR
9556 052560 105777 126760 7$: TSTB @TKS ::CHAR THERE?
9557 052564 100375 BPL 7$ ::IF NOT TRY AGAIN
9558
9559 052566 117746 126754 MOVB @TKB,-(SP) ::PICK UP CHAR
9560 052572 042716 177600 BIC #^C177,(SP) ::MAKE IT 7-BIT ASCII
9561
9562
9563
9564 052576 021627 000025 9$: CMP (SP),#25 ::IS IT A CONTROL-U?
9565 052602 001005 BNE 10$ ::BRANCH IF NOT
9566 052604 104401 053206 TYPE ,SCNTLU ::YES, ECHO CONTROL-U (^U)
9567 052610 062706 000006 20$: ADD #6,SP ::IGNORE PREVIOUS INPUT
9568 052614 000757 BR 19$ ::LET'S TRY IT AGAIN
9569
9570
9571 052616 021627 000015 10$: CMP (SP),#15 ::IS IT A <CR>?
9572 052622 001022 BNE 16$ ::BRANCH IF NO
9573 052624 005766 000004 TST 4(SP) ::YES, IS IT THE FIRST CHAR?
9574 052630 001403 BEQ 11$ ::BRANCH IF YES
9575 052632 016677 000002 126700 MOV 2(SP),@SWR ::SAVE NEW SWR
9576 052640 062706 000006 11$: ADD #6,SP ::CLEAR UP STACK
9577 052644 104401 001713 14$: TYPE ,$CRLF ::ECHO <CR> AND <LF>
9578 052650 123727 001535 000001 CMPB $INTAG,#1 ::RE-ENABLE TTY KBD INTERRUPTS?
9579 052656 001003 BNE 15$ ::BRANCH IF NOT
9580 052660 012777 000100 126656 MOV #100,@TKS ::RE-ENABLE TTY KBD INTERRUPTS
9581 052666 000002 15$: RTI ::RETURN
9582 052670 004737 053550 16$: JSR PC,$TYPEC ::ECHO CHAR
9583 052674 021627 000060 CMP (SP),#60 ::CHAR < 0?
9584 052700 002420 BLT 18$ ::BRANCH IF YES
9585 052702 021627 000067 CMP (SP),#67 ::CHAR > 7?
9586 052706 003015 BGT 18$ ::BRANCH IF YES
9587 052710 042726 000060 BIC #60,(SP)+ ::STRIP-OFF ASCII
9588 052714 005766 000002 TST 2(SP) ::IS THIS THE FIRST CHAR
    
```

```

9589 052720 001403          BEQ      17$          ::BRANCH IF YES
9590 052722 006316          ASL      (SP)         ::NO, SHIFT PRESENT
9591 052724 006316          ASL      (SP)         ::  CHAR OVER TO MAKE
9592 052726 006316          ASL      (SP)         ::  ROOM FOR NEW ONE.
9593 052730 005266 000002   17$:  INC      2(SP)         ::KEEP COUNT OF CHAR
9594 052734 056616 177776   BIS      -2(SP),(SP)  ::SET IN NEW CHAR
9595 052740 000707          BR       7$          ::GET THE NEXT ONE
9596 052742 104401 001712   18$:  TYPE    $QUES      ::TYPE ?<CR><LF>
9597 052746 000720          BR       20$         ::SIMULATE CONTROL-U
9598
9599
9600
9601
9602
9603
9604
9605
9606
9607
9608
9609 052750 011646          $RDCHR: MOV      (SP),-(SP)  ::PUSH DOWN THE PC
9610 052752 016666 000004 000002   1$:  MOV      4(SP),2(SP)  ::SAVE THE PS
9611 052760 105777 126560   TSTB    @STKS         ::WAIT FOR
9612 052764 100375          BPL      1$          ::A CHARACTER
9613 052766 117766 126554 000004   MOVB    @STKB,4(SP)   ::READ THE TTY
9614 052774 042766 177600 000004   BIC     #'C<177>,4(SP) ::GET RID OF JUNK IF ANY
9615 053002 026627 000004 000023   CMP     4(SP),#23    ::IS IT A CONTROL-S?
9616 053010 001013          BNE     3$          ::BRANCH IF NO
9617 053012 105777 126526   2$:  TSTB    @STKS         ::WAIT FOR A CHARACTER
9618 053016 100375          BPL     2$          ::LOOP UNTIL ITS THERE
9619 053020 117746 126522   MOVB    @STKB,-(SP)  ::GET CHARACTER
9620 053024 042716 177600   BIC     #'C177,(SP)  ::MAKE IT 7-BIT ASCII
9621 053030 022627 000021   CMP     (SP)+,#21    ::IS IT A CONTROL-Q?
9622 053034 001366          BNE     2$          ::IF NOT DISCARD IT
9623 053036 000750          BR      1$          ::YES, RESUME
9624 053040 026627 000004 000140   3$:  CMP     4(SP),#140   ::IS IT UPPER CASE?
9625 053046 002407          BLT     4$          ::BRANCH IF YES
9626 053050 026627 000004 000175   CMP     4(SP),#175   ::IS IT A SPECIAL CHAR?
9627 053056 003003          BGT     4$          ::BRANCH IF YES
9628 053060 042766 000040 000004   BIC     #40,4(SP)    ::MAKE IT UPPER CASE
9629 053066 000002          4$:  RTI                    ::GO BACK TO USER
9630
9631
9632
9633
9634
9635
9636
9637 053070 010346          $RDLIN: MOV     R3,-(SP)  ::SAVE R3
9638 053072 012703 053176   1$:  MOV     #$TTYIN,R3   ::GET ADDRESS
9639 053076 022703 053206   2$:  CMP     #$TTYIN+8.,R3 ::BUFFER FULL?
9640 053102 101405          BLOS    4$          ::BR IF YES
9641 053104 104410          RDCHR   ::GO READ ONE CHARACTER FROM THE TTY
9642 053106 112613          MOVB    (SP)+,(R3)   ::GET CHARACTER
9643 053110 122713 000177   10$:  CMPB    #177,(R3)    ::IS IT A RUBOUT
9644 053114 001003          BNE     3$          ::SKIP IF NOT
    
```

\*\*\*\*\*  
 \*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

\*CALL:  
 \* RDCHR ::INPUT A SINGLE CHARACTER FROM THE TTY  
 \* RETURN HERE ::CHARACTER IS ON THE STACK  
 \* ::WITH PARITY BIT STRIPPED OFF

\$RDCHR: MOV (SP),-(SP) ::PUSH DOWN THE PC  
 1\$: MOV 4(SP),2(SP) ::SAVE THE PS  
 TSTB @STKS ::WAIT FOR  
 BPL 1\$ ::A CHARACTER  
 MOVB @STKB,4(SP) ::READ THE TTY  
 BIC #'C<177>,4(SP) ::GET RID OF JUNK IF ANY  
 CMP 4(SP),#23 ::IS IT A CONTROL-S?  
 BNE 3\$ ::BRANCH IF NO  
 2\$: TSTB @STKS ::WAIT FOR A CHARACTER  
 BPL 2\$ ::LOOP UNTIL ITS THERE  
 MOVB @STKB,-(SP) ::GET CHARACTER  
 BIC #'C177,(SP) ::MAKE IT 7-BIT ASCII  
 CMP (SP)+,#21 ::IS IT A CONTROL-Q?  
 BNE 2\$ ::IF NOT DISCARD IT  
 BR 1\$ ::YES, RESUME  
 3\$: CMP 4(SP),#140 ::IS IT UPPER CASE?  
 BLT 4\$ ::BRANCH IF YES  
 CMP 4(SP),#175 ::IS IT A SPECIAL CHAR?  
 BGT 4\$ ::BRANCH IF YES  
 BIC #40,4(SP) ::MAKE IT UPPER CASE  
 4\$: RTI ::GO BACK TO USER

\*\*\*\*\*  
 \*THIS ROUTINE WILL INPUT A STRING FROM THE TTY

\*CALL:  
 \* RDLIN ::INPUT A STRING FROM THE TTY  
 \* RETURN HERE ::ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK  
 \* ::TERMINATOR WILL BE A BYTE OF ALL 0'S

\$RDLIN: MOV R3,-(SP) ::SAVE R3  
 1\$: MOV #\$TTYIN,R3 ::GET ADDRESS  
 2\$: CMP #\$TTYIN+8.,R3 ::BUFFER FULL?  
 BLOS 4\$ ::BR IF YES  
 RDCHR ::GO READ ONE CHARACTER FROM THE TTY  
 MOVB (SP)+,(R3) ::GET CHARACTER  
 10\$: CMPB #177,(R3) ::IS IT A RUBOUT  
 BNE 3\$ ::SKIP IF NOT

```

9645 053116 104401 001712 4$: TYPE $QUES ::TYPE A '?'
9646 053122 000763 BR 1$ ::CLEAR THE BUFFER AND LOOP
9647 053124 111337 053174 3$: MOVB (R3),9$ ::ECHO THE CHARACTER
9648 053130 104401 053174 TYPE 9$
9649 053134 122723 000015 CMPB #15,(R3)+ ::CHECK FOR RETURN
9650 053140 001356 BNE 2$ ::LOOP IF NOT RETURN
9651 053142 105063 177777 CLRB -1(R3) ::CLEAR RETURN (THE 15)
9652 053146 104401 001714 TYPE $LF ::TYPE A LINE FEED
9653 053152 012603 MOV (SP)+,R3 ::RESTORE R3
9654 053154 011646 MOV (SP),-(SP) ::ADJUST THE STACK AND PUT ADDRESS OF THE
9655 053156 016666 000004 000002 MOV 4(SP),2(SP) :: FIRST ASCII CHARACTER ON IT
9656 053164 012766 053176 000004 MOV #$TTYIN,4(SP)
9657 053172 000002 RTI ::RETURN
9658 053174 000 9$: .BYTE 0 ::STORAGE FOR ASCII CHAR. TO TYPE
9659 053175 000 .BYTE 0 ::TERMINATOR
9660 053176 000010 $TTYIN: .BLKB 8. ::RESERVE 8 BYTES FOR TTY INPUT
9661 053206 052536 005015 000 $CNTLU: .ASCIZ /^U/<15><12> ::CONTROL 'U'
9662 053213 136 006507 000012 $CNTLG: .ASCIZ /^G/<15><12> ::CONTROL 'G'
9663 053220 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
9664 053226 020075 000 $MNEW: .ASCIZ / NEW = /
9665 053231 040 047040 053505
9666 053236 036440 000040
    
```

.SBTTL SAVE AND RESTORE R0-R5 ROUTINES

```

::*****
:*SAVE R0-R5
:*CALL:
:* SAVREG
:*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
:*
:*TOP---(+16)
:* +2---(+18)
:* +4---R5
:* +6---R4
:* +8---R3
:*+10---R2
:*+12---R1
:*+14---R0
    
```

```

$SAVREG:
MOV R0,-(SP) ::PUSH R0 ON STACK
MOV R1,-(SP) ::PUSH R1 ON STACK
MOV R2,-(SP) ::PUSH R2 ON STACK
MOV R3,-(SP) ::PUSH R3 ON STACK
MOV R4,-(SP) ::PUSH R4 ON STACK
MOV R5,-(SP) ::PUSH R5 ON STACK
MOV 22(SP),-(SP) ::SAVE PS OF MAIN FLOW
MOV 22(SP),-(SP) ::SAVE PC OF MAIN FLOW
MOV 22(SP),-(SP) ::SAVE PS OF CALL
MOV 22(SP),-(SP) ::SAVE PC OF CALL
RTI
    
```

```

:*RESTORE R0-R5
:*CALL:
    
```

9667  
9668  
9669  
9670  
9671  
9672  
9673  
9674  
9675  
9676  
9677  
9678  
9679  
9680  
9681  
9682  
9683  
9684  
9685  
9686 053242  
9687 053242 010046  
9688 053244 010146  
9689 053246 010246  
9690 053250 010346  
9691 053252 010446  
9692 053254 010546  
9693 053256 016646 000022  
9694 053262 016646 000022  
9695 053266 016646 000022  
9696 053272 016646 000022  
9697 053276 000002  
9698  
9699  
9700

```

9701          : *      RESREG
9702 053300    $RESREG:
9703 053300    012666 000022    MOV      (SP)+,22(SP)    ;;RESTORE PC OF CALL
9704 053304    012666 000022    MOV      (SP)+,22(SP)    ;;RESTORE PS OF CALL
9705 053310    012666 000022    MOV      (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
9706 053314    012666 000022    MOV      (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
9707 053320    012605          MOV      (SP)+,R5        ;;POP STACK INTO R5
9708 053322    012604          MOV      (SP)+,R4        ;;POP STACK INTO R4
9709 053324    012603          MOV      (SP)+,R3        ;;POP STACK INTO R3
9710 053326    012602          MOV      (SP)+,R2        ;;POP STACK INTO R2
9711 053330    012601          MOV      (SP)+,R1        ;;POP STACK INTO R1
9712 053332    012600          MOV      (SP)+,R0        ;;POP STACK INTO R0
9713 053334    000002          RTI
9714
9715          .SBTTL  TYPE ROUTINE
9716
9717          ;*****
9718          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
9719          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
9720          ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
9721          ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
9722          ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
9723          ;*
9724          ;*CALL:
9725          ;*1) USING A TRAP INSTRUCTION
9726          ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
9727          ;*OR
9728          ;*      TYPE
9729          ;*      MESADR
9730          ;*
9731
9732 053336    105737 001557    $TYPE:  TSTB      $TPFLG      ;;IS THERE A TERMINAL?
9733 053342    100002          BPL      1$              ;;BR IF YES
9734 053344    000000          HALT          ;;HALT HERE IF NO TERMINAL
9735 053346    000430          BR      3$              ;;LEAVE
9736 053350    010046          1$:  MOV      RO,-(SP)      ;;SAVE RO
9737 053352    017600 000002    MOV      @2(SP),RO      ;;GET ADDRESS OF ASCIZ STRING
9738 053356    122737 000001 001736    CMPB     #APTENV,$ENV    ;;RUNNING IN APT MODE
9739 053364    001011          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
9740 053366    132737 000100 001737    BITB     #APTPOOL,$ENVM  ;;SPOOL MESSAGE TO APT
9741 053374    001405          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
9742 053376    010037 053406    MOV      RO,61$         ;;SETUP MESSAGE ADDRESS FOR APT
9743 053402    004737 052226    JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
9744 053406    000000          61$:  .WORD     0           ;;MESSAGE ADDRESS
9745 053410    132737 000040 001737    62$:  BITB     #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
9746 053416    001003          BNE     60$             ;;YES,SKIP TYPE OUT
9747 053420    112046          2$:  MOVB     (RO)+,-(SP)   ;;PUSH CHARACTER TO BE TYPED ONTO STACK
9748 053422    001005          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
9749 053424    005726          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
9750 053426    012600          60$:  MOV      (SP)+,RO      ;;RESTORE RO
9751 053430    062716 000002    3$:  ADD      #2,(SP)        ;;ADJUST RETURN PC
9752 053434    000002          RTI
9753 053436    122716 000011    4$:  CMPB     #HT,(SP)       ;;BRANCH IF <HT>
9754 053442    001430          BEQ     8$              ;;
9755 053444    122716 000200    CMPB     #CRLF,(SP)     ;;BRANCH IF NOT <CRLF>
9756 053450    001006          BNE     5$
    
```

```
9757 053452 005726      TST      (SP)+      ;;POP <CR><LF> EQUIV
9758 053454 104401      TYPE                                ;;TYPE A CR AND LF
9759 053456 001713      $CRLF                                ;;
9760 053460 105037 053614  CLR      $CHARCNT    ;;CLEAR CHARACTER COUNT
9761 053464 000755      BR      2$          ;;GET NEXT CHARACTER
9762 053466 004737 053550 5$:      JSR      PC,$TYPEC  ;;GO TYPE THIS CHARACTER
9763 053472 123726 001556 6$:      CMPB     $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
9764 053476 001350      BNE     2$          ;;IF NO GO GET NEXT CHAR.
9765 053500 013746 001554      MOV     $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
9766                                     ;;AND THE NULL CHAR.
9767 053504 105366 000001 7$:      DECB     1(SP)      ;;DOES A NULL NEED TO BE TYPED?
9768 053510 002770      BLT     6$          ;;BR IF NO--GO POP THE NULL OFF OF STACK
9769 053512 004737 053550      JSR     PC,$TYPEC  ;;GO TYPE A NULL
9770 053516 105337 053614      DECB     $CHARCNT   ;;DO NOT COUNT AS A COUNT
9771 053522 000770      BR      7$          ;;LOOP
9772
9773                                     ;HORIZONTAL TAB PROCESSOR
9774
9775 053524 112716 000040 8$:      MOV     #' ,(SP)   ;;REPLACE TAB WITH SPACE
9776 053530 004737 053550 9$:      JSR     PC,$TYPEC  ;;TYPE A SPACE
9777 053534 132737 000007 053614  BITB     #',$CHARCNT ;;BRANCH IF NOT AT
9778 053542 001372      BNE     9$          ;;TAB STOP
9779 053544 005726      TST     (SP)+      ;;POP SPACE OFF STACK
9780 053546 000724      BR      2$          ;;GET NEXT CHARACTER
9781 053550 105777 125774 $TYPEC: TSTB     @$TPS    ;;WAIT UNTIL PRINTER IS READY
9782 053554 100375      BPL     $TYPEC     ;;
9783 053556 116677 000002 125766  MOV     2(SP),@$TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
9784 053564 122766 000015 000002  CMPB     #CR,2(SP)  ;;IS CHARACTER A CARRIAGE RETURN?
9785 053572 001003      BNE     1$          ;;BRANCH IF NO
9786 053574 105037 053614  CLR      $CHARCNT   ;;YES--CLEAR CHARACTER COUNT
9787 053600 000406      BR      $TYPEX     ;;EXIT
9788 053602 122766 000012 000002 1$:      CMPB     #LF,2(SP) ;;IS CHARACTER A LINE FEED?
9789 053610 001402      BEQ     $TYPEX     ;;BRANCH IF YES
9790 053612 105227      INCB     (PC)+     ;;COUNT THE CHARACTER
9791 053614 000000 $CHARCNT: .WORD 0   ;;CHARACTER COUNT STORAGE
9792 053616 000207 $TYPEX:  RTS      PC
9793
9794
9795                                     .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
9796
9797                                     ;*****
9798                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
9799                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
9800                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
9801                                     ;*CALL:
9802                                     ;*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
9803                                     ;*      TYPOS                                ;;CALL FOR TYPEOUT
9804                                     ;*      .BYTE  N                      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
9805                                     ;*      .BYTE  M                      ;;M=1 OR 0
9806                                     ;*                                     ;;1=TYPE LEADING ZEROS
9807                                     ;*                                     ;;0=SUPPRESS LEADING ZEROS
9808                                     ;*
9809                                     ;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
9810                                     ;*$TYPOS OR $TYPOC
9811                                     ;*CALL:
9812                                     ;*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
```



9869 054042 000  
9870 054043 000  
9871 054044 000000

\$OCNT: .BYTE 0 ::OCTAL DIGIT COUNTER  
\$OFILL: .BYTE 0 ::ZERO FILL SWITCH  
\$OMODE: .WORD 0 ::NUMBER OF DIGITS TO TYPE

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

9872  
9873  
9874  
9875  
9876  
9877  
9878  
9879  
9880  
9881  
9882  
9883  
9884

\*\*\*\*\*  
\*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT  
\*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE  
\*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED  
\*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE  
\*REPLACED WITH SPACES.  
\*CALL:

\* MOV NUM,-(SP) ::PUT THE BINARY NUMBER ON THE STACK  
\* TYPDS ::GO TO THE ROUTINE

9885 054046  
9886 054046 010046  
9887 054050 010146  
9888 054052 010246  
9889 054054 010346  
9890 054056 010546  
9891 054060 012746 020200  
9892 054064 016605 000020  
9893 054070 100004  
9894 054072 005405  
9895 054074 112766 000055 000001  
9896 054102 005000  
9897 054104 012703 054262  
9898 054110 112723 000040  
9899 054114 005002  
9900 054116 016001 054252  
9901 054122 160105  
9902 054124 002402  
9903 054126 005202  
9904 054130 000774  
9905 054132 060105  
9906 054134 005702  
9907 054136 001002  
9908 054140 105716  
9909 054142 100407  
9910 054144 106316  
9911 054146 103003  
9912 054150 116663 000001 177777  
9913 054156 052702 000060  
9914 054162 052702 000040  
9915 054166 110223  
9916 054170 005720  
9917 054172 020027 000010  
9918 054176 002746  
9919 054200 003002  
9920 054202 010502  
9921 054204 000764  
9922 054206 105726  
9923 054210 100003  
9924 054212 116663 177777 177776

\$TYPDS: MOV R0,-(SP) ::PUSH R0 ON STACK  
MOV R1,-(SP) ::PUSH R1 ON STACK  
MOV R2,-(SP) ::PUSH R2 ON STACK  
MOV R3,-(SP) ::PUSH R3 ON STACK  
MOV R5,-(SP) ::PUSH R5 ON STACK  
MOV #20200,-(SP) ::SET BLANK SWITCH AND SIGN  
MOV 20(SP),R5 ::GET THE INPUT NUMBER  
BPL 1\$ ::BR IF INPUT IS POS.  
NEG R5 ::MAKE THE BINARY NUMBER POS.  
MOVB #'-,1(SP) ::MAKE THE ASCII NUMBER NEG.  
1\$: CLR R0 ::ZERO THE CONSTANTS INDEX  
MOV #SDBLK,R3 ::SETUP THE OUTPUT POINTER  
MOVB #' ,(R3)+ ::SET THE FIRST CHARACTER TO A BLANK  
2\$: CLR R2 ::CLEAR THE BCD NUMBER  
MOV \$DTBL(R0),R1 ::GET THE CONSTANT  
3\$: SUB R1,R5 ::FORM THIS BCD DIGIT  
BLT 4\$ ::BR IF DONE  
INC R2 ::INCREASE THE BCD DIGIT BY 1  
BR 3\$  
4\$: ADD R1,R5 ::ADD BACK THE CONSTANT  
TST R2 ::CHECK IF BCD DIGIT=0  
BNE 5\$ ::FALL THROUGH IF 0  
TSTB (SP) ::STILL DOING LEADING 0'S?  
BMI 7\$ ::BR IF YES  
5\$: ASLB (SP) ::MSD?  
BCC 6\$ ::BR IF NO  
MOVB 1(SP),-1(R3) ::YES--SET THE SIGN  
6\$: BIS #'0,R2 ::MAKE THE BCD DIGIT ASCII  
7\$: BIS #' ,R2 ::MAKE IT A SPACE IF NOT ALREADY A DIGIT  
MOVB R2,(R3)+ ::PUT THIS CHARACTER IN THE OUTPUT BUFFER  
TST (R0)+ ::JUST INCREMENTING  
CMP R0,#10 ::CHECK THE TABLE INDEX  
BLT 2\$ ::GO DO THE NEXT DIGIT  
BGT 8\$ ::GO TO EXIT  
MOV R5,R2 ::GET THE LSD  
BR 6\$ ::GO CHANGE TO ASCII  
8\$: TSTB (SP)+ ::WAS THE LSD THE FIRST NON-ZERO?  
BPL 9\$ ::BR IF NO  
MOVB -1(SP),-2(R3) ::YES--SET THE SIGN FOR TYPING

9925 054220 105013  
 9926 054222 012605  
 9927 054224 012603  
 9928 054226 012602  
 9929 054230 012601  
 9930 054232 012600  
 9931 054234 104401 054262  
 9932 054240 016666 000002 000004  
 9933 054246 012616  
 9934 054250 000002  
 9935 054252 023420  
 9936 054254 001750  
 9937 054256 000144  
 9938 054260 000012  
 9939 054262 000004  
 9940  
 9941  
 9942  
 9943  
 9944  
 9945  
 9946  
 9947  
 9948  
 9949  
 9950  
 9951  
 9952 054272  
 9953 054272 010046  
 9954 054274 010146  
 9955 054276 010246  
 9956 054300 013700 054372  
 9957 054304 013701 054370  
 9958 054310 012702 177771  
 9959 054314 006300  
 9960 054316 006101  
 9961 054320 005202  
 9962 054322 001374  
 9963 054324 063700 054372  
 9964 054330 005501  
 9965 054332 063701 054370  
 9966 054336 062700 001057  
 9967 054342 005501  
 9968 054344 062701 047401  
 9969 054350 010037 054372  
 9970 054354 010137 054370  
 9971 054360 012602  
 9972 054362 012601  
 9973 054364 012600  
 9974 054366 000207  
 9975 054370 176543  
 9976 054372 123456  
 9977  
 9978  
 9979  
 9980

```

9$:  CLR8    (R3)          ;;SET THE TERMINATOR
      MOV    (SP)+,R5      ;;POP STACK INTO R5
      MOV    (SP)+,R3      ;;POP STACK INTO R3
      MOV    (SP)+,R2      ;;POP STACK INTO R2
      MOV    (SP)+,R1      ;;POP STACK INTO R1
      MOV    (SP)+,R0      ;;POP STACK INTO R0
      TYPE   $DBLK         ;;NOW TYPE THE NUMBER
      MOV    2(SP),4(SP)   ;;ADJUST THE STACK
      MOV    (SP)+,(SP)
      RTI                    ;;RETURN TO USER
    
```

\$DTBL: 10000.  
 1000.  
 100.  
 10.

\$DBLK: .BLKW 4

.SBTTL RANDOM NUMBER GENERATOR ROUTINE

```

*****
*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
*WITH A RANGE OF 0 TO 2(+33)-1.
*CALL:
*      JSR    PC,$RAND     ;;CALL THE ROUTINE
*      RETURN              ;;RETURN HERE THE RANDOM
*                          ;;NUMBER WILL BE IN
*                          ;;$HINUM,$LONUM
    
```

```

$RAND:
      MOV    R0,-(SP)      ;;PUSH R0 ON STACK
      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
      MOV    R2,-(SP)      ;;PUSH R2 ON STACK
      MOV    $LONUM,R0     ;;SET R0 WITH LOW
      MOV    $HINUM,R1     ;;SET R1 WITH HIGH
      MOV    #-7,R2        ;;SET SHIFT COUNT
1$:   ASL    R0              ;;SHIFT R0 LEFT AND
      ROL    R1              ;;ROTATE CARRY INTO R1 AND
      INC    R2              ;;CHECK FOR DONE
      BNE    1$             ;;CONTINUE SHIFT LOOP
      ADD    $LONUM,R0      ;;ADD NUMBER TO MAKE X 129
      ADC    R1              ;;PROPOGATE CARRY
      ADD    $HINUM,R1      ;;ADD NUMBER TO MAKE X 129
      ADD    #1057,R0       ;;ADD LOW CONSTANT
      ADC    R1              ;;PROPOGATE CARRY
      ADD    #47401,R1      ;;ADD HIGH CONSTANT
      MOV    R0,$LONUM     ;;SAVE R0
      MOV    R1,$HINUM     ;;SAVE R1
      MOV    (SP)+,R2      ;;POP STACK INTO R2
      MOV    (SP)+,R1      ;;POP STACK INTO R1
      MOV    (SP)+,R0      ;;POP STACK INTO R0
      RTS    PC            ;;RETURN
    
```

\$HINUM: .WORD 176543  
 \$LONUM: .WORD 123456

.SBTTL TRAP DECODER

\*\*\*\*\*



9981  
 9982  
 9983  
 9984  
 9985  
 9986 054374 010046  
 9987 054376 016600 000002  
 9988 054402 005740  
 9989 054404 111000  
 9990 054406 006300  
 9991 054410 016000 054430  
 9992 054414 000200  
 9993  
 9994  
 9995  
 9996  
 9997 054416 011646  
 9998 054420 016666 000004 000002  
 9999 054426 000002  
 10000  
 10001  
 10002  
 10003  
 10004  
 10005  
 10006  
 10007  
 10008 054430 054416  
 10009 054432 053336  
 10010 054434 053644  
 10011 054436 053620  
 10012 054440 053660  
 10013 054442 054046  
 10014  
 10015 054444 052536  
 10016  
 10017 054446 052466  
 10018 054450 052750  
 10019 054452 053070  
 10020 054454 053242  
 10021 054456 053300  
 10022  
 10023 054460 055574  
 10024 054462 055544  
 10025 054464 056256  
 10026 054466 056300  
 10027 054470 055664  
 10028 054472 055710  
 10029 054474 055726  
 10030 054476 055744  
 10031 054500 055762  
 10032 054502 041752  
 10033  
 10034 054504 061046  
 10035 054506 060076  
 10036 054510 063542

;\*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION  
 ;\*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS  
 ;\*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL  
 ;\*GO TO THAT ROUTINE.

```
$TRAP:  MOV    R0,-(SP)      ;;SAVE R0
        MOV    2(SP),R0     ;;GET TRAP ADDRESS
        TST    -(R0)        ;;BACKUP BY 2
        MOVB   (R0),R0      ;;GET RIGHT BYTE OF TRAP
        ASL    R0           ;;POSITION FOR INDEXING
        MOV    $TRPAD(R0),R0 ;;INDEX TO TABLE
        RTS    R0           ;;GO TO ROUTINE
```

;;THIS IS USE TO HANDLE THE 'GETPRI' MACRO

```
$TRAP2: MOV    (SP),-(SP)   ;;MOVE THE PC DOWN
        MOV    4(SP),2(SP)  ;;MOVE THE PSW DOWN
        RTI                    ;;RESTORE THE PSW
```

.SBTTL TRAP TABLE

;\*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED  
 ;\*BY THE 'TRAP' INSTRUCTION.

ROUTINE	\$TRAP2	TRAP+X(ADDRESS)	DESCRIPTION
\$.WORD	\$TRAP2	TRAP+1(104401)	TTY TYPEOUT ROUTINE
\$TYPE	::CALL=TYPE	TRAP+2(104402)	TYPE OCTAL NUMBER (WITH LEADING ZEROS)
\$TYPOC	::CALL=TYPOC	TRAP+3(104403)	TYPE OCTAL NUMBER (NO LEADING ZEROS)
\$TYPOS	::CALL=TYPOS	TRAP+4(104404)	TYPE OCTAL NUMBER (AS PER LAST CALL)
\$TYPON	::CALL=TYPON	TRAP+5(104405)	TYPE DECIMAL NUMBER (WITH SIGN)
\$TYPDS	::CALL=TYPDS		
\$GTSWR	::CALL=GTSWR	TRAP+6(104406)	GET SOFT-SWR SETTING
\$CKSWR	::CALL=CKSWR	TRAP+7(104407)	TEST FOR CHANGE IN SOFT-SWR
\$RDCHR	::CALL=RDCHR	TRAP+10(104410)	TTY TYPEIN CHARACTER ROUTINE
\$RDLIN	::CALL=RDLIN	TRAP+11(104411)	TTY TYPEIN STRING ROUTINE
\$SAVREG	::CALL=SAVREG	TRAP+12(104412)	SAVE R0-R5 ROUTINE
\$RESREG	::CALL=RESREG	TRAP+13(104413)	RESTORE R0-R5 ROUTINE
CLEAN	::CALL=RSET	TRAP+14(104414)	GO RESET ALL REGISTERS.
ABORTT	::CALL=SKIPT	TRAP+15(104415)	THIS WILL SKIP TO THE NEXT TEST
MMDDES	::CALL=MMSKIP	TRAP+16(104416)	IF SWITCH # IS ON SKIP TO THE NEXT TEST
MSIZER	::CALL=SIZE	TRAP+17(104417)	DETERMINE THE HIGHEST ADDRESS IN MEMORY
SKBADR	::CALL=SKPBAD	TRAP+20(104420)	SKIP TEST IF ERROR ADDRESS REGISTER IS I
SKBERR	::CALL=SKPBER	TRAP+21(104421)	SKIP TEST IF ERROR REGISTER IS INOPERATI
SKBCNR	::CALL=SKPBCN	TRAP+22(104422)	SKIP TEST IF CONTROL REGISTER IS INOPERA
SKBMNR	::CALL=SKPBMN	TRAP+23(104423)	SKIP TEST IF MAINTENANCE REGISTER IS INO
SKBHMR	::CALL=SKPBHM	TRAP+24(104424)	SKIP TEST IF HIT/MISS REGISTER IS IN OPE
RANDWR	::CALL=WRRAND	TRAP+25(104425)	FILL BUFFER WITH RANDOM SEQUENCE
RS4HAN	::CALL=CALRS4	TRAP+26(104426)	DO RS04 FUNCTION
RP4HAN	::CALL=CALRP4	TRAP+27(104427)	DO RPO4 FUNCTION
RH4HAN	::CALL=CALRH4	TRAP+30(104430)	DO MBT FUNCTION

```

10037 054512 062002          RKSHAN  ::CALL=CALRK5  TRAP+31(104431) DO RK05 FUNCTION
10038 054514 063016          UBEHAN  ::CALL=CALUBE  TRAP+32(104432) DO UBE FUNCTION
10039                                     .SBTTL  POWER DOWN AND UP ROUTINES
10040
10041                                     ::*****
10042                                     :POWER DOWN ROUTINE
10043 054516 012737 054662 000024 $PWRDN: MOV    #$ILLUP,@#PWRVEC  ::SET FOR FAST UP
10044 054524 012737 000340 000026      MOV    #340,@#PWRVEC+2  ::PRIO:7
10045 054532 010046          MOV    R0,-(SP)         ::PUSH R0 ON STACK
10046 054534 010146          MOV    R1,-(SP)         ::PUSH R1 ON STACK
10047 054536 010246          MOV    R2,-(SP)         ::PUSH R2 ON STACK
10048 054540 010346          MOV    R3,-(SP)         ::PUSH R3 ON STACK
10049 054542 010446          MOV    R4,-(SP)         ::PUSH R4 ON STACK
10050 054544 010546          MOV    R5,-(SP)         ::PUSH R5 ON STACK
10051 054546 017746 124766          MOV    @SWR,-(SP)      ::PUSH @SWR ON STACK
10052 054552 010637 054666          MOV    SP,$SAVR6      ::SAVE SP
10053 054556 012737 054570 000024      MOV    #$PWRUP,@#PWRVEC ::SET UP VECTOR
10054 054564 000000          HALT
10055 054566 000776          BR     .-2             ::HANG UP
10056
10057                                     ::*****
10058                                     :POWER UP ROUTINE
10059 054570 012737 054662 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC  ::SET FOR FAST DOWN
10060 054576 013706 054666          MOV    $SAVR6,SP      ::GET SP
10061 054602 005037 054666          CLR    $SAVR6        ::WAIT LOOP FOR THE TTY
10062 054606 005237 054666          1$:  INC    $SAVR6     ::WAIT FOR THE INC
10063 054612 001375          BNE    1$             ::OF WORD
10064 054614 012677 124720          MOV    (SP)+,@SWR     ::POP STACK INTO @SWR
10065 054620 012605          MOV    (SP)+,R5      ::POP STACK INTO R5
10066 054622 012604          MOV    (SP)+,R4      ::POP STACK INTO R4
10067 054624 012603          MOV    (SP)+,R3      ::POP STACK INTO R3
10068 054626 012602          MOV    (SP)+,R2      ::POP STACK INTO R2
10069 054630 012601          MOV    (SP)+,R1      ::POP STACK INTO R1
10070 054632 012600          MOV    (SP)+,R0      ::POP STACK INTO R0
10071 054634 012737 054516 000024      MOV    #$PWRDN,@#PWRVEC ::SET UP THE POWER DOWN VECTOR
10072 054642 012737 000340 000026      MOV    #340,@#PWRVEC+2 ::PRIO:7
10073 054650 104401          TYPE                                ::REPORT THE POWER FAILURE
10074 054652 066433          $PWRMG: .WORD  POWERM              ::POWER FAIL MESSAGE POINTER
10075 054654 012716          MOV    (PC)+,(SP)    ::RESTART AT START
10076 054656 004146          $PWRAD: .WORD  START              ::RESTART ADDRESS
10077 054660 000002          RTI
10078 054662 000000          $ILLUP: HALT          ::THE POWER UP SEQUENCE WAS STARTED
10079 054664 000776          BR     .-2             :: BEFORE THE POWER DOWN WAS COMPLETE
10080 054666 000000          $SAVR6: 0              ::PUT THE SP HERE
10081                                     .SBTTL  ROUTINE TO SIZE MEMORY
10082
10083                                     ::*****
10084                                     :CALL:
10085                                     *      JSR    PC,$SIZE
10086                                     *      RETURN
10087                                     *$LSTAD WILL CONTAIN:
10088                                     *      WITH KT11 OPTION          -- LAST VIRTUAL ADDRESS OF THE LAST BANK
10089                                     *      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
10090                                     *$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
10091                                     *$KT11 IS THE MEMORY MANAGEMENT KEY
10092                                     *BIT07 = 0 DON'T USE MEMORY MANAGEMENT
    
```

10093  
10094  
10095  
10096  
10097  
10098  
10099  
10100  
10101  
10102  
10103  
10104  
10105  
10106  
10107  
10108  
10109  
10110  
10111  
10112  
10113  
10114  
10115  
10116  
10117  
10118  
10119  
10120  
10121  
10122  
10123  
10124  
10125  
10126  
10127  
10128  
10129  
10130  
10131  
10132  
10133  
10134  
10135  
10136  
10137  
10138  
10139  
10140  
10141  
10142  
10143  
10144  
10145  
10146  
10147  
10148

054670 010046  
054672 010146  
054674 010246  
054676 010346  
054700 013746 000114  
054704 013746 000116  
054710 012737 000116 000114  
054716 012737 000002 000116  
054724 013746 000004  
054730 013746 000006  
054734 010600  
054736 104400  
054740 012637 000006  
054744 012701 003776  
054750 105727  
054752 000200  
054754 100065  
054756 012737 055122 000004  
054764 005737 177572  
054770 052737 100000 054752  
054776 005046  
055000 012702 172340  
055004 012703 000010  
055010 012762 077406 177740 1\$:  
055016 011622  
055020 062716 000200  
055024 077307  
055026 012742 177600  
055032 005042  
055034 012737 055052 000004  
055042 012737 000020 172516  
055050 000401  
055052 022626 2\$:  
055054 005237 177572 3\$:  
055060 012737 055112 000004  
055066 012737 055234 000114  
055074 005737 143776 4\$:  
055100 062712 000040  
055104 022712 170000  
055110 101371  
055112 011202  
055114 005037 177572  
055120 000421

```

;* MUST BE SETUP BEFORE THE CALL
;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
;* DETERMINED BY ROUTINE
;* --NOTE--
;*THIS ROUTINE SUPPORTS PDP 11/74.
;*IF ACTUAL MEMORY IS LESS THAN THAT INDICATED BY THE SIZE REGISTER
;*AND A REFERENCE IS MADE TO A MEMORY ADDRESS THAT IS GREATER THAN
;*ACTUAL MEMORY BUT LESS THAN SIZE REGISTER (INDICATED), THEN A
;*MEMORY REFERENCE TIMEOUT TO VECTOR 114 WILL OCCUR.
;* --NOTE--
;*THIS ROUTINE WILL NOT SIZE FOR MEMORY GREATER THAN 1920K.

$SIZE:  MOV R0,-(SP)      ;;SAVE R0 ON THE STACK
        MOV R1,-(SP)      ;;SAVE R1 ON THE STACK
        MOV R2,-(SP)      ;;SAVE R2 ON THE STACK
        MOV R3,-(SP)      ;;SAVE R3 ON THE STACK
        MOV @#114,-(SP)    ;;SAVE MEMORY ERROR VECTOR PS & PC
        MOV @#116,-(SP)    ;;
        MOV #116,@#114     ;;IGNORE PARITY ERRORS WHILE SIZING
        MOV #RTI,@#116     ;;
        MOV @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
        MOV @#ERRVEC+2,-(SP) ;;
        MOV SP,R0          ;;SAVE THE STACK POINTER
        ;;SET THE ERRVEC PS TO THE PRESENT PS
        TRAP              ;;PUSH OLD PSW AND PC ON STACK
        MOV (SP)+,@#ERRVEC+2 ;;SAVE THE PSW IN @#ERRVEC+2
        MOV #3776,R1      ;;SETUP ADDRESS
        TSTB (PC)+        ;;USE MEMORY MANAGEMENT?
$KT11:  .WORD 200         ;;SET TO USE MEMORY MANAGEMENT
        BPL $SCORE        ;;BR IF NO
        MOV #SKTNEX,@#ERRVEC ;;SET FOR TIMEOUT
        TST @#SR0         ;;KT11 ARE YOU THERE?
        BIS #100000,$KT11 ;;YES--SET KT11 KEY
        CLR -(SP)         ;;INITIALIZE FOR 'PAR' LOADING
        MOV #KIPAR0,R2    ;;ADDRESS OF FIRST 'PAR'
        MOV #^D8,R3       ;;LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
        MOV #77406,-40(R2) ;;PDR = 4K, UP, READ/WRITE
        MOV (SP),(R2)+    ;;LOAD 'PAR'
        ADD #200,(SP)     ;;UPDATE FOR NEXT 'PAR'
        SOB R3,1$        ;;LOOP UNTIL ALL EIGHT ARE LOADED
        MOV #177600,-(R2) ;;SETUP KIPAR7 FOR I/O
        CLR -(R2)         ;;SETUP KIPAR6 FOR TESTING
        MOV #2$,@#ERRVEC  ;;CATCH TIMEOUT IF NO SR3
        MOV #20,@#SR3     ;;ENABLE 22 BIT MODE
        BR 3$             ;;THIS PDP-11 HAS A SR3 REGISTER
        CMP (SP)+,(SP)+  ;;CLEAN OFF THE STACK--NO SR3
        INC @#SR0         ;;TURN ON MEMORY MANAGEMENT
        MOV #SKTOUT,@#ERRVEC ;;SET FOR TIME OUT
        MOV #SMTMOUT,@#114 ;;SET FOR MEM REF TIMEOUT
        TST @#143776      ;;TRAP ON NON-EX-MEM
        ADD #40,(R2)     ;;MAKE A 1K STEP
        CMP #170000,(R2) ;;LAST ONE?
        BHI 4$           ;;NO--TRY IT
$SKTOUT: MOV (R2),R2     ;;GET LAST BANK+1
        CLR @#SR0        ;;TURN OFF MEMORY MANAGEMENT
        BR $SIZEX
    
```

(REV F)  
(REV F)  
(REV F)  
(REV F)

```

10149 055122 042737 100000 054752 $KTNEX: BIC #100000,$KT11 ::KT11 NON-EXISTENT
10150 055130 012737 055160 000004 $CORC: MOV # $CROUT,@#ERRVEC ::SET FOR TIMEOUT
10151 055136 005002 CLR R2 ::SET UP BANK
10152 055140 062701 004000 1$: ADD #4000,R1 ::INCREMENT BY 1K
10153 055144 062702 000040 ADD #40,R2 ::1K STEP
10154 055150 005711 TST (R1) ::TRAP ON TIME OUT
10155 055152 022701 177776 CMP #177776,R1 ::LAST ONE
10156 055156 001370 BNE 1$ ::NO--TRY AGAIN
10157 055160 162701 004000 $CROUT: SUB #4000,R1
10158 055164 162702 000040 $SIZE: SUB #40,R2 ::DROP BACK
10159 055170 010006 MOV R0,SP ::RESTORE THE STACK
10160 055172 012637 000006 MOV (SP)+,@#ERRVEC+2 ::RESTORE ERROR VECTOR
10161 055176 012637 000004 MOV (SP)+,@#ERRVEC
10162 055202 012637 000116 MOV (SP)+,@#116 ::RESTORE MEMORY ERROR VECTOR
10163 055206 012637 000114 MOV (SP)+,@#114
10164 055212 010137 055266 MOV R1,$LSTAD ::LAST ADDRESS
10165 055216 010237 055270 MOV R2,$LSTBK ::LAST BANK
10166 055222 012603 MOV (SP)+,R3 ::RESTORE R3
10167 055224 012602 MOV (SP)+,R2 ::RESTORE R2
10168 055226 012601 MOV (SP)+,R1 ::RESTORE R1
10169 055230 012600 MOV (SP)+,R0 ::RESTORE R0
10170 055232 000207 RTS PC
10171 055234 032737 000001 177744 $MTMOUT: BIT #BIT0,@#MEMERR ::MAKE SURE TRAP TO 114 IS DUE
10172 055242 001005 BNE 1$ ::TO MEMORY REFERENCE TIMEOUT
10173 BIT #BIT15,@#MEMERR ::IF NOT, IS IT AN ABORT?
10174 055244 032737 100000 177744 BIT #BIT15,@#MEMERR ::CPU ABORT?
10175 055252 001001 BNE 1$ ::IF YES, EXIT OUT
10176 055254 000002 RTI ::IF NOT, CONTINUE
10177 055256 012737 177777 177744 1$: MOV #-1,@#MEMERR ::CLEAR THE MEM ERROR REG
10178 055264 000712 BR $KTOUT
10179 055266 000000 $LSTAD: .WORD 0 ::CONTAINS THE LAST ADDRESS
10180 055270 000000 $LSTBK: .WORD 0 ::CONTAINS THE LAST BANK
10181
10182 .SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
10183
10184 ::*****
10185 ::*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
10186 ::*UNSIGNED OCTAL ASCII NUMBER.
10187 ::*CALL
10188 ::* MOV #PNTR,-(SP) ::*POINTER TO LOW WORD OF BINARY NUMBER
10189 ::* JSR PC,@#$DB20 ::*CALL THE ROUTINE
10190 ::* RETURN ::*THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK
10191
10192
10193 055272 104412 $DB20: SAVREG ::SAVE ALL REGISTERS
10194 055274 016601 000002 MOV 2(SP),R1 ::PICKUP THE POINTER TO LOW WORD
10195 055300 012705 055411 MOV # $OCTVL+13.,R5 ::POINTER TO DATA TABLE
10196 055304 012704 000014 MOV #12.,R4 ::DO ELEVEN CHARACTERS
10197 055310 012703 177770 MOV #^C7,R3 ::MASK
10198 055314 012100 MOV (R1)+,R0 ::LOWER WORD
10199 055316 012101 MOV (R1)+,R1 ::HIGH WORD
10200 055320 005002 CLR R2 ::TERMINATOR
10201 055322 110245 1$: MOV R2,-(R5) ::PUT CHARACTER IN DATA TABLE
10202 055324 010002 MOV R0,R2 ::GET THIS DIGIT
10203 055326 005304 DEC R4 ::COUNT THIS CHARACTER
10204 055330 003007 BGT 3$ ::BR IF NOT THE LAST DIGIT

```

(REV F)  
(REV F)

```

10205 055332 001405          BEQ      2$          ;;BR IF IT IS THE LAST DIGIT
10206 055334 005205          INC      R5          ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
10207 055336 010566 000002  MOV      R5,2(SP)   ;;ASCIZ CHAR. & PUT IT ON THE STACK
10208 055342 104413          RESREG          ;;RESTORE ALL REGISTERS
10209 055344 000207          RTS      PC          ;;RETURN TO USER
10210 055346 006203 2$:    ASR      R3          ;;POSITION THE MASK FOR THE LAST DIGIT
10211 055350 006001 3$:    ROR      R1          ;;POSITION THE BINARY NUMBER FOR
10212 055352 006000          ROR      R0          ;;
10213 055354 006001          ROR      R1          ;;
10214 055356 006000          ROR      R0          ;;
10215 055360 006001          ROR      R1          ;;
10216 055362 006000          ROR      R0          ;;
10217 055364 040302          BIC      R3,R2      ;;MASK OUT ALL JUNK
10218 055366 062702 000060  ADD      #'0,R2     ;;MAKE THIS CHAR. ASCII
10219 055372 000753          BR       1$          ;;GO PUT IT IN THE DATA TABLE
10220 055374 000016          $OCTVL: .BLKB 14.  ;;RESERVE DATA TABLE
    
```

```

10221
10222          ;THIS ROUTINE IS CALLED BY UNEXPECTED TRAPS TO VECTOR ERRVEC.
10223          ;THE ERROR IS REPORTED AND CONTROL IS TRANSFERRED BACK TO THE TEST
10224          ;FOLLOWING THE ONE THAT WAS INTERRUPTED WHEN THE ERROR OCCURRED!
10225 055412 011637 001634  CPSPUR: MOV      (SP), $TMP1
10226 055416 012737 055434 001636  MOV      #1$, $TMP2
10227 055424 013737 177766 001640  MOV      @#CPUERR, $TMP3
10228 055432 022626          CMP      (SP)+, (SP)+ ;RESET THE STACK
10229 055434 104150 1$:    ERROR  150
10230 055436 104415          SKIPT
    
```

```

10231
10232          ;THIS ROUTINE HANDLE UNEXPECTED TRAPS TO #CACHVEC.
10233 055440 012737 055536 000114  SPUR:  MOV      #10$, @#CACHVEC
10234 055446 013700 177744          MOV      @#MEMERR, R0
10235 055452 032700 000014          BIT      #14, R0      ;SEE IF IT WAS A MAIN MEMORY ERROR.
10236 055456 001405          BEQ      9$
10237 055460 013701 177740          MOV      @#LOADRS, R1 ;IF SO THERE IS BAD PARITY IN THE
10238 055464 042701 176000          BIC      #176000, R1 ;CACHE AND IT MUST BE PURGED!!!
10239 055470 005711          TST      (R1)
10240 055472 012737 055440 000114  9$:    MOV      #SPUR, @#CACHVEC
10241 055500 013737 177744 001642  MOV      @#MEMERR, $TMP4 ;TRAP HERE IF AN UNEXPECTED
10242 055506 013737 177740 001634  MOV      @#LOADRS, $TMP1 ;ERROR, PARITY, OCCURS.
10243 055514 013737 177742 001636  MOV      @#HIADRS, $TMP2
10244 055522 011637 001640          MOV      (SP), $TMP3
10245 055526 022626          CMP      (SP)+, (SP)+
10246 055530 104014 1$:    ERROR  14
10247 055532 000005          RESET
10248 055534 104415          SKIPT          ;TO STOP THE ACTION OF ANY I/O DEVICE!!!!
10249 055536 022626          10$:    CMP      (SP)+, (SP)+ ;?????
10250 055540 000137 055472          JMP      9$
    
```

```

10251
10252          ;THIS ROUTINE IS CALLED BY THE TRAP CATCHER CALL SKIPT.
10253          ;IT TELLS THE USER THAT THE CURRENT TEST HAS BEEN
10254          ;ABORTED AND THAT CONTROL IS BEING PASSED TO THE NEXT TEST.
10255 055544 011637 001634  ABORTT: MOV      (SP), $TMP1
10256 055550 112737 000015 001514  MOVB     #15, $ITEMB
10257 055556 022626          CMP      (SP)+, (SP)+
10258 055560 004737 056354          JSR      PC, ERTYPE
10259 055564 104414          RSET
10260 055566 000177 000000          JMP      @SKAD      ;GO TO @SKAD, WHICH SHOULD
    
```

```

10261                                     ;BE SET TO THE
10262 055572 000000 SKAD: .WORD 0          ;ADDRESS OF THE NEXT TEST.
10263
10264
10265                                     ;THIS ROUTINE IS CALLED BY THE TRAP CATCHER CALL RSET. IT CLEARS ALL
10266                                     ;THE IMPORTANE REGISTERS AND RESETS THE STACK.
10267 055574 CLEAN:
10268
10269 055574 012737 055440 000114          MOV    #SPUR,@#CACHVEC
10270 055602 012737 055412 000004          MOV    #CPSPUR,@#ERRVEC
10271 055610 011637 055662                MOV    (SP),BACKAD
10272 055614 012706 001400                MOV    #STACK,SP
10273 055620 005037 177750                CLR    @#MAINT          ;CLEAR ALL CONTROL AND ERROR
10274 055624 005037 177572                CLR    @#MMR0          ;REGISTERS.
10275 055630 005037 172516                CLR    @#MMR3
10276 055634 005037 177746                CLR    @#CONTRL
10277 055640 012737 177777 177744          MOV    #-1,@#MEMERR
10278 055646 005037 177766                CLR    @#CPUERR
10279 055652 005037 177776                CLR    @#PSW
10280 055656 000177 000000                JMP    @BACKAD
10281 055662 000000 BACKAD: .WORD 0
10282
10283
10284                                     ;COME HERE TO TEST THE REGISTER FLAGS AND USE THEM TO DETERMINE WHETHER
10285                                     ;OR NOT TO SKIP A TEST WHICH RELIES ON THE FUNCTIONALLITY OF THAT REGISTER
10286                                     ;TO BE PROPERLY RUN.
10287                                     ;THESE ROUTINES ARE CALLED BY THE TRAP CATCHER CALLS:
10288                                     : SKPBAD          SKIPT IF BAD ERROR ADDRESS REGISTER
10289                                     : SKPBER          SKIPT IF BAD ERROR REGISTER
10290                                     : SKPBCN          SKIPT IF BAD CONTROL REGISTER
10291                                     : SKPBMM          SKIPT IF BAD MAINTENANCE REGISTER
10292                                     : SKPBHM          SKIPT IF BAD HIT/MISS REGISTER
10293                                     :
10294
10295 055664 005737 056002 SKBADR: TST    LOAFLG
10296 055670 001004          BNE    1$
10297 055672 005737 056004          TST    HIAFLG
10298 055676 001001          BNE    1$
10299 055700 000002          RTI
10300 055702 104401          1$:    TYPE
10301 055704 067415          .WORD  ADRNG
10302 055706 000433          BR     SKRNG
10303
10304 055710 005737 056006 SKBERR: TST    MMRFLG
10305 055714 001001          BNE    1$
10306 055716 000002          RTI
10307 055720 104401          1$:    TYPE
10308 055722 067525          .WORD  ERRNG
10309 055724 000424          BR     SKRNG
10310
10311 055726 005737 056010 SKBCNR: TST    CONFLG
10312 055732 001001          BNE    1$
10313 055734 000002          RTI
10314 055736 104401          1$:    TYPE
10315 055740 067625          .WORD  CNRNG
10316 055742 000415          BR     SKRNG
    
```

10317  
 10318 055744 005737 056012  
 10319 055750 001001  
 10320 055752 000002  
 10321 055754 104401  
 10322 055756 067727  
 10323 055760 000406  
 10324  
 10325 055762 005737 056014  
 10326 055766 001001  
 10327 055770 000002  
 10328 055772 104401  
 10329 055774 070035  
 10330  
 10331 055776 022626  
 10332 056000 104415  
 10333  
 10334 056002 000000  
 10335 056004 000000  
 10336 056006 000000  
 10337 056010 000000  
 10338 056012 000000  
 10339 056014 000000  
 10340 056016 000000  
 10341 056020 000000  
 10342 056022 000000  
 10343 056024 000000  
 10344 056026 000000  
 10345 056030 000000  
 10346  
 10347  
 10348  
 10349  
 10350  
 10351  
 10352  
 10353  
 10354  
 10355 056032 012701 000001  
 10356 056036 005002  
 10357 056040 030100  
 10358 056042 001401  
 10359 056044 005202  
 10360 056046 006301  
 10361 056050 103373  
 10362 056052 000207  
 10363  
 10364  
 10365  
 10366  
 10367  
 10368  
 10369  
 10370  
 10371  
 10372

SKBMNR: TST MANFLG  
 BNE 1\$  
 RTI  
 1\$: TYPE  
 .WORD MNRNG  
 BR SKRNG

SKBHMR: TST HIMFLG  
 BNE 1\$  
 RTI  
 1\$: TYPE  
 .WORD HMRNG

SKRNG: CMP (SP)+,(SP)+  
 SKIPT

LOAFLG: .WORD 0  
 HIAFLG: .WORD 0  
 MMRFLG: .WORD 0  
 CONFLG: .WORD 0  
 MANFLG: .WORD 0  
 HIMFLG: .WORD 0  
 LOAFL2: .WORD 0  
 HIAFL2: .WORD 0  
 MMRFL2: .WORD 0  
 CONFL2: .WORD 0  
 MANFL2: .WORD 0  
 HIMFL2: .WORD 0

;RESET THE STACK AND GO TO THE  
 ;NEXT TEST!!!!

;THESE ARE FLAGS USED TO DESIGNATE  
 ;EITHER A GOOD OR A BAD REGISTER.  
 ;GOOD WILL BE DESIGNATED BY A  
 ;0 BAD BY A NOT ZERO!!

;THIS ROUTINE IS CALLED TO DETERMINE THE PARITY OF  
 ;A DATA PATTERN. THE PATTERN WHICH IS TAKEN BY THIS  
 ;ROUTINE AS ITS ARGUMENT SHOULD BE PUT IN R0. THEN  
 ;TRANSFER CONTROL HERE BY EXECUTING:  
 ; JSR PC,PARCNT  
 ;WHEN THIS ROUTINE RETURNS THE NUMBER OF ON,(1), BITS  
 ;IN R0 IS LEFT IN R2. THIS WOULD BE A NUMBER BETWEEN  
 ;0 AND 16.

PARCNT: MOV #1,R1  
 CLR R2  
 1\$: BIT R1,R0  
 BEQ 2\$  
 INC R2  
 2\$: ASL R1  
 BCC 1\$  
 RTS PC

;THIS ROUTINE IS CALLED TO RESTORE THE TOP 1500 (DEC) WORDS IN THE  
 ;FIRST 28K OF MEMORY. THIS SHOULD EFFECTIVELY RESTORE ANY MONITOR  
 ;OR LOADER THAT WAS PRESENT BEFORE THIS PROGRAM BEGAN EXECUTION.  
 ;CONTROL IS PASSED TO THIS ROUTINE BY AN INTERRUPT FROM THE TTY KEYBOARD  
 ;WHEN ANY CHARACTER IS TYPED ON THE KEYBOARD. IF THE CHARACTER  
 ;IS A ^C MEMORY IS RESTORED. IF IT'S A ^G THE PROGRAM REQUESTS  
 ;A NEW SOFTWARE SWR.  
 ; A RETURN IS MADE TO THE TEST FOLLOWING  
 ;THE ONE WHOSE EXECUTION WAS INTERRUPTED BY THE KEYBOARD INTERRUPT.

10373	056054	005037	177750		RESMON:	CLR	@#MAINT	
10374	056060	017700	123462			MOV	@\$TKB,R0	
10375	056064	104414				RSET		
10376	056066	005003				CLR	R3	
10377	056070	042700	177600			BIC	#177600,R0	:GET THE CHARACTER
10378	056074	022700	000003			CMP	#3,R0	:SEE IF IT'S ^C
10379	056100	001037				BNE	NOCNC	:BRANCH AND GO TO NEXT TEST IF NOT.
10380	056102	104401				TYPE		:ECHO THE CONTROL-C AS '^C'
10381	056104	066370				.WORD	CONCMS	
10382	056106	012704	002734		CHAINQ:	MOV	#^D1500,R4	:AND RESTORE THE MONITOR.
10383	056112	012701	120314			MOV	#BOTTOM+4,R1	
10384	056116	012702	160000			MOV	#160000,R2	
10385	056122	012142			1\$:	MOV	(R1)+,-(R2)	
10386	056124	077402				SOB	R4,1\$	
10387	056126	012737	177777	056254		MOV	#-1,MONF	:RESET THE MONITOR RESTORED FLAG.
10388	056134	022703	125252			CMP	#125252,R3	
10389	056140	001001				BNE	STOP	
10390	056142	000207				RTS	PC	:IF THE MONITOR WAS RESTORED BY THE
10391								:.SEOP ROUTIN RETURN TO .SEOP.
10392								:OTHERWISE HALT.
10393	056144	005737	000042		STOP:	TST	@#42	:IN AUTO MODE?
10394	056150	001402				BEQ	1\$	:BRANCH IF NOT
10395	056152	000137	051454			JMP	\$GET42+16	:ELSE RETURN TO MONITOR
10396	056156	104401			1\$:	TYPE		:TYPE THE MONITOR RESTORED MESSAGE.
10397	056160	066374				.WORD	MMSRS	
10398	056162	013737	056252	000060		MOV	MONTTY,@#TKVEC	:SET THE TTY KEYBOARD INTERRUPT VECTOR
10399								:TO ITS INITIAL VALUE.
10400	056170	000000				HALT		:AND HALT!!
10401	056172	012737	056054	000060		MOV	#RESMON,@#TKVEC	
10402	056200	022737	000176	001540	NOCNC:	CMP	#SWREG,SWR	:SOFTWARE SWR SELECTED?
10403	056206	001012				BNE	1\$	:BRANCH IF NOT
10404	056210	022700	000007			CMP	#7,R0	:IS IT ^G?
10405	056214	001007				BNE	1\$	:BRANCH IF NOT
10406	056216	123727	001534	000001		CMPB	\$AUTOB,#1	:ARE WE RUNNING IN AUTO MODE?
10407	056224	001403				BEQ	1\$	:BRANCH IF YES
10408	056226	104401	053213			TYPE	,\$CNTLG	:ECHO ^G
10409	056232	104406				GTSWR		:GET NEW SWR SETTING
10410	056234	005077	123306		1\$:	CLR	@\$TKB	:NOT CONTROL C SO RETURN TO NEXT TEST.
10411	056240	152777	000100	123276		BISB	#BIT6,@\$TKS	
10412	056246	000177	177320			JMP	@SKAD	:RETURN.
10413	056252	000000			MONTTY:	.WORD	0	:STORAGE FOR THE TTY KEYBOARD VECTOR'S ORIGINAL
10414								:CONTENTS.
10415	056254	177777			MONF:	.WORD	177777	:FLAG. IF NOT -1 THE MONITOR IS SAVED.
10416								
10417								
10418								
10419								
10420								
10421								
10422								
10423								
10424								
10425								
10426								
10427	056256	032777	000200	123254	MMDDES:	BIT	#SW7,@SWR	
10428	056264	001001				BNE	1\$	:IS THE SWITCH ON?

:THIS ROUTINE IS CALLED BY THE TRAP CALL MMSKIP. IT LOOKS  
 :AT THE SWITCH REGISTER AND DETERMINES WHETHER OR NOT  
 :SWITCH #7 IS ON. IF SO THE CURRENT TEST IS SKIPPED  
 :AND THE NEXT TEST IS ENTERED. A SSKAD MUST BE ISSUED  
 :BEFORE THE MMSKIP.  
 :THE PURPOSE OF SWITCH #7 IS TO CAUSE THE DELETION OF THE  
 :EXECUTION OF ANY TEST WHICH RELIES ON MEMORY MANAGEMENT  
 :FOR ITS OPERATION.



10429 056266 000002  
 10430 056270 022626  
 10431 056272 104414  
 10432 056274 000177 177272  
 10433  
 10434  
 10435  
 10436  
 10437  
 10438  
 10439  
 10440  
 10441  
 10442  
 10443 056300 010046  
 10444 056302 010146  
 10445 056304 016600 000004  
 10446 056310 013710 055270  
 10447 056314 005060 000002  
 10448 056320 012701 000006  
 10449  
 10450 056324 006310  
 10451 056326 006160 000002  
 10452 056332 077104  
 10453 056334 052710 000076  
 10454  
 10455  
 10456 056340 022020  
 10457 056342 010066 000004  
 10458  
 10459 056346 012601  
 10460 056350 012600  
 10461 056352 000002  
 10462  
 10463  
 10464  
 10465  
 10466  
 10467 056354 104401  
 10468 056356 001713  
 10469 056360 010046  
 10470 056362 005000  
 10471 056364 113700 001514  
 10472 056370 001005  
 10473 056372 013746 001516  
 10474 056376 104402  
 10475 056400 000137 056716  
 10476  
 10477 056404 005300  
 10478 056406 072027 000003  
 10479 056412 062700 001752  
 10480 056416 012037 056426  
 10481 056422 001404  
 10482 056424 104401  
 10483 056426 000000  
 10484 056430 104401

```

RTI                                ;NO, SO RETURN.
1$:  CMP      (SP)+,(SP)+
     RSET
     JMP      @SKAD                ;YES, GO TO THE NEXT TEST.
;THIS ROUTINE IS CALLED TO DETERMINE THE HIGHEST POSSIBLE
;ADDRESS IN MEMORY. IT IS CALLED THUS, BY TRAP CALL SIZE:
;
;      SIZE
;      LOORDA: .WORD  0
;      HIORDA: .WORD  0
;      NXTINST:
;THE LOW ORDER 16-BITS OF THE ADDRESS ARE LEFT IN THE
;WORD DIRECTLY FOLLOWING THE CALL. THE HIGH ORDER 6-BITS
;ARE LEFT IN THE NEXT WORD AND CONTROL IS RETURNED
;TO THE THIRD WORD FOLLOWING THE CALL.
MSIZER: MOV    R0,-(SP)           ;SAVE THE CONTENTS OF R0 AND R1
        MOV    R1,-(SP)           ;GET THE ADDRESS OF
        MOV    4(SP),R0           ;THE CALL OF THE STACK.
        MOV    $LSTBK,(R0)       ;GET THE ACCESSABLE BOUNDARY OF MEMORY
        CLR    2(R0)
        MOV    #6,R1
;ROTATE THE 16-BIT 'BLOCK'
;NUMBER 6-BITS TO THE
1$:  ASL      (R0)                ;LEFT AND TURN ON LOW ORDER
     ROL      2(R0)              ;BITS 1-5 LEAVING BIT-0
     SOB      R1,1$              ;OFF SO AS TO CREATE
     BIS      #76,(R0)           ;THE 22-BIT PHYSICAL ADDRESS OF
;THE HIGHEST WORD IN
;MEMORY.
;DETERMINE THE RETURN ADDRESS
;AND LEAVE ON THE STACK FOR
;AN RTI.
        MOV    (SP)+,R1          ;RESTORE R1 AND R0.
        MOV    (SP)+,R0
        RTI                        ;RETURN
;THIS ROUTINE IS USED TO TYPE AN ERROR MESSAGE
;WHICH IS IN THE DATA TABLE. IT IS CALLED BY
;THE $ERROR ROUTINE OR BY FIRST SETTING THE $ITEMB
;BYTE EQUAL TO THE ERROR TABLE ITEM NUMBER THAT IS
;TO BE PRINTED OUT AND THEN EXECUTING A JSR PC,ERTYPE
ERTYPE: TYPE
        .WORD  $CRLF
        MOV    R0,-(SP)           ;SAVE R0
        CLR    R0
        MOVB   $ITEMB,R0         ;GET THE ITEM NUMBER
        BNE    1$                 ;ZERO?
        MOV    $ERRPC,-(SP)       ;YES, TYPE JUST THE PC
;OF THE ERROR CALL.
        JMP    ERT5
1$:  DEC      R0                  ;MAKE R0 AN INDEX FOR THE
     ASH      #3,R0              ;ERROR TABLE
     ADD      #$ERRTB,R0
     MOV      (R0)+,2$           ;TYPE EM, ERROR MESSAGE.
     BEQ     3$
     TYPE
2$:  .WORD  0
     TYPE
    
```

```

10485 056432 001713
10486 056434 012037 056444 3$: .WORD $CRLF
MOV (R0)+,4$ ;TYPE DH, DATA HEADER
BEQ 5$
10487 056440 001404
10488 056442 104401
10489 056444 000000 4$: .WORD 0
TYPE
10490 056446 104401
10491 056450 001713 5$: .WORD $CRLF
MOV R1,-(SP) ;SAVE R1
MOV (R0)+,R1 ;GET DT, DATA TABLE ADDRESS
BNE 6$
10494 056456 001002
10495 056460 000137 056714 6$: MOV (R0)+,R0 ;JMP IF NO ERROR TABLE.
ERT1: TSTB (R0) ;GET DF, DATA FORMAT ADDRESS
BNE 7$ ;DATA FORMAT ENTRY EQUALS
MOV @ (R1)+,-(SP) ;ZERO?
;YES, SO TYPE A 16-BIT
;OCTAL NUMBER
TYPOC
BR ERT2
10501 056476 000500
10502 056500 122710 000001 7$: CMPB #1,(R0) ;FORMAT EQUALS 1?
BNE 8$
MOV @ (R1)+,-(SP) ;YES, TYPE A DECIMAL NUMBER
TYPDS
BR ERT2
10506 056512 000472
10507
10508 056514 122710 000002 8$: CMPB #2,(R0) ;FORMAT 2?
BNE 9$
10509 056520 001012
10510 056522 012146 85$: MOV (R1)+,-(SP) ;YES, TYPE A 22-BIT NUMBR
JSR PC,$DB20 ;CALL $DB20 TO CONVERT THE
ADD #3,(SP) ;BINARY TO ASCII
MOV (SP)+,29$ ;TYPE THE STRING
10513 056534 012637 056542
10514 056540 104401
10515 056542 000000 29$: .WORD 0
BR ERT2
10516 056544 000455
10517
10518 056546 122710 000004 9$: CMPB #4,(R0) ;FORMAT 4?
BNE 10$
MOV @ (R1)+,-(SP) ;YES, TYPE A 16-BIT
;OCTAL NUMBER SUPRESSING
;LEADING ZEROES
TYPOS
.BYTE 16
.BYTE 0
BR ERT2
10524 056562 000446
10525 056564 122710 000003 10$: CMPB #3,(R0) ;FORMAT 3?
BNE 11$
MOV @ (R1)+,-(SP) ;YES CONVERT 16-BIT
;VIRTUAL ADDRESS TO 32-BIT
;PHYSICAL ADDRESS AND TYPE
;RELOCATE ONLY IF SEG. IS ON!
10527 056572 013146
10528 056574 012737 177777 056722
10529 056602 004737 056730
10530 056606 000434
10531 056610 122710 000005 11$: CMPB #5,(R0) ;FORMAT 5?
BNE 12$
MOV (R1)+,20$ ;PRINT ASCII STRING
10532 056614 001005
10533 056616 012137 056624
10534 056622 104401
10535 056624 000000 20$: .WORD 0
BR ERT3
10536 056626 000426
10537
10538 056630 122710 000006 12$: CMPB #6,(R0) ;FORMAT 6
BNE 13$
10539 056634 001005
10540 056636 005037 056722 CLR TVADFL
    
```



```

10597 057000 006001          ROR    R1          ;OF THE VIRTUAL ADDRESS
10598 057002 006001          ROR    R1          ;TO THE PAR
10599 057004 062700 172340   ADD    #KIPARO,R0  ;DETERMINE THE CORRECT PAR'S
10600                                ;ADDRESS
10601 057010 011003          MOV    (R0),R3     ;GET ITS CONTENTS
10602 057012 005002          CLR    R2
10603 057014 073227 000006   ASHC  #6,R2       ;MAKE THE BLOCK COUNT
10604                                ;A 22-BIT ADDRESS.
10605 057020 060103          ADD    R1,R3      ;ADD THE OFFSET TO THE
10606 057022 005502          ADC    R2         ;BASE ADDRESS
10607
10608 057024 010237 056726   MOV    R2,TVADHI
10609 057030 010337 056724   MOV    R3,TVADLO
10610 057034 012746 056724   2$:   MOV    #TVADLO,-(SP) ;CALL $DB20 TO CONVERT THE
10611 057040 004737 055272   JSR    PC,$DB20   ;22-BIT
10612 057044 062716 000003   ADD    #3,(SP)   ;TYPE ONLY 8 DIGITS.
10613 057050 012637 057056   MOV    (SP)+,3$
10614 057054 104401
10615 057056 000000          3$:   .WORD 0
10616 057060 104413          RESREG
10617 057062 012616          MOV    (SP)+,(SP) ;RESTORE THE REGISTERS
10618                                ;LEAVE ONLY THE RETURN
10619 057064 000207          RTS    PC         ;ADDRESS ON THE STACK.
10620                                ;RETURN
10621                                .SBTTL SYSTEM DEVICE SIZER
10622
10623                                ;THIS ROUTINE IS CALLED TO DETERMINE WHAT
10624                                ;CONTROLLERS AND WHAT DRIVES ARE AVAILABLE ON
10625                                ;THE SYSTEM.
10626                                ;IT USES THE FLAGS:
10627                                ;
10628                                ;
10629                                ;
10630                                ;
10631                                ;
10632                                ;
10633                                ;
10634                                ;
10635                                ;
10636                                ;
10637                                ;
10638                                ;
10639                                ;
10640                                ;
10641                                ;
10642                                ;
10643                                ;
10644                                ;
10645                                ;
10646                                ;
10647                                ;
10648                                ;
10649                                ;
10650                                ;
10651                                ;
10652                                ;
10634 057066 005037 061002   SIZDEV: CLR    RS4FLG ;INITIALIZE FLAG
10635 057072 005037 060032   CLR    RP4FLG
10636 057076 005037 063476   CLR    RH4FLG
10637 057102 005037 061736   CLR    RK5FLG
10638 057106 005037 062752   CLR    UBFLG
10639 057112 005037 061004   CLR    RS4ER1
10640 057116 005037 060034   CLR    RP4ER1
10641 057122 005037 063500   CLR    RH4ER1
10642 057126 005037 061740   CLR    RK5ER1
10643 057132 005037 062754   CLR    UBER1
10644 057136 104412          SAVREG
10645 057140 105037 057460   CLRB  RS4DFL
10646 057144 105037 057461   CLRB  RP4DFL
10647 057150 105037 057462   CLRB  RH4DFL
10648 057154 105037 057463   CLRB  RK5DFL
10649 057160 105037 057464   CLRB  UBDFL
10650
10651 057164 013737 000004 057466   MOV    @#4,SIZTM1 ;SAVE 4
10652 057172 012737 057220 000004   MOV    #1,@#4     ;IN CASE NON-EXISTENT REG.
    
```



```

10709 057440 022626          11$:  CMP      (SP)+,(SP)+      ;THERE WAS NO UBE
10710 057442 005037 177766      CLR      @#CPUERR
10711
10712 057446 013737 057466 000004 12$:  MOV      SIZTM1,@#4      ;RESTORE 4
10713 057454 104413      RESREG
10714 057456 000207      RTS      PC
10715
10716 057460      000      RS4DFL: .BYTE 0
10717 057461      000      RP4DFL: .BYTE 0
10718 057462      000      RH4DFL: .BYTE 0
10719 057463      000      RK5DFL: .BYTE 0
10720 057464      000      UBEDFL: .BYTE 0
10721      057466      .EVEN
10722
10723 057466 000000      SIZTM1: .WORD 0
10724 057470 000000      SIZTM2: .WORD 0
10725 057472 000000      SIZTM3: .WORD 0
10726 057474 000000      SIZTM4: .WORD 0
10727 057476 000000      SIZTM5: .WORD 0
10728
10729      ;THIS ROUTINE IS CALED BY A:
10730      ;JSR      PC,SETREG
10731      ;.WORD  DEVREG
10732      ;WHERE DEVREG IS THE STARTING ADDRESS OF
10733      ;A TABLE, WHICH IS TO CONTAIN THE ADDRESS OF
10734      ;A DEVICE'S CONTROL AND STATUS REGISTERS.
10735      ;THE TABLES ARE GENERATE HERE
10736
10737 057500 011637 057536      SETREG: MOV      (SP),SETMP
10738 057504 062716 000002      ADD      #2,(SP)
10739 057510 104412      SAVREG
10740 057512 017700 000020      MOV      @SETMP,R0
10741 057516 012001      MOV      (R0)+,R1
10742 057520 011002      MOV      (R0),R2
10743 057522 010220      1$:  MOV      R2,(R0)+
10744 057524 062702 000002      ADD      #2,R2
10745 057530 077104      SOB      R1,1$
10746 057532 104413      RESREG
10747 057534 000207      RTS      PC
10748
10749 057536 000000      SETMP: .WORD 0
10750
10751
10752      ;THIS ROUTINE IS CALLED, AFTER IT HAS BEEN
10753      ;DETERMINED IF THERE IS A RS04 CONTROLLER, TO SEE
10754      ;WHT DRIVES ARE AVAILABLE.
10755
10756 057540 012700 000010      SIZRS4: MOV      #10,R0
10757
10758 057544 012701 000001      MOV      #1,R1
10759 057550 005002      CLR      R2
10760 057552 105037 057561      CLRB     3$
10761
10762 057556 104426      1$:  CALRS4
10763 057560      001      2$:  .BYTE 1
10764 057561      000      3$:  .BYTE 0      ;DO A NOP FUNCTION
    
```

```

10765 057562 000000          .WORD 0          ;FOR EACH OF POSSIBLY
10766 057564 000000          .WORD 0          ;8 DRIVES
10767 057566 000000          .WORD 0
10768 057570 000000          .WORD 0
10769 057572 000000          .WORD 0
10770 057574 000000          .WORD 0
10771
10772 057576 005737 061004    TST      RS4ER1
10773 057602 001001          BNE      4$
10774 057604 050102          BIS      R1,R2
10775 057606 006301          4$:     ASL      R1
10776 057610 105237 057561    INCB     3$
10777 057614 077020          SOB      R0,1$
10778
10779 057616 110237 057460    MOVB     R2,RS4DFL
10780 057622 000207          RTS      PC
10781
10782          ;THIS ROUTINE IS CALLED TO DETERMINE WHAT RP04
10783          ;DRIVES ARE ON THE CONTROLLER
10784
10785 057624 012700 000010    SIZRP4: MOV      #10,R0
10786 057630 012701 000001    MOV      #1,R1
10787 057634 005002          CLR      R2
10788 057636 105037 057645    CLRB     3$
10789
10790 057642 104427          1$:     CALRP4
10791 057644      021          2$:     .BYTE 21          ;DO A READ IN PRESET
10792 057645      000          3$:     .BYTE 0          ;FOR EACH OF UP TO
10793 057646 000000          .WORD 0          ;8 DRIVES.
10794 057650 000000          .WORD 0
10795 057652 000000          .WORD 0
10796 057654 000000          .WORD 0
10797 057656 000000          .WORD 0
10798 057660 000000          .WORD 0
10799
10800 057662 005737 060034    TST      RP4ER1
10801 057666 001001          BNE      4$
10802 057670 050102          BIS      R1,R2
10803 057672 006301          4$:     ASL      R1
10804 057674 105237 057645    INCB     3$
10805 057700 077020          SOB      R0,1$
10806
10807 057702 110237 057461    MOVB     R2,RP4DFL
10808 057706 000207          RTS      PC
10809
10810          ;DETERMINE WHAT RK05 DRIVES ARE AVAILABLE.
10811
10812 057710 012700 000010    SIZRK5: MOV      #10,R0
10813 057714 012701 000001    MOV      #1,R1
10814 057720 005002          CLR      R2
10815
10816 057722 105037 057731    CLRB     3$
10817
10818
10819 057726 104431          1$:     CALRK5
10820 057730      015          .BYTE 15          ;DO A DRIVE RESET
                    ;FOR EACH OF 8
    
```

```

10821 057731 000 3$: .BYTE 0 ;POSSIBLE DRIVES.
10822 057732 000000 .WORD 0
10823 057734 000000 .WORD 0
10824 057736 000000 .WORD 0
10825 057740 000000 .WORD 0
10826 057742 000000 .WORD 0
10827 057744 000000 .WORD 0
    
```

```

10828
10829 057746 005737 061740 TST RK5ER1
10830 057752 001001 BNE 4$
10831 057754 050102 BIS R1,R2
10832 057756 006301 4$: ASL R1
10833 057760 105237 057731 INCB 3$
10834 057764 077020 SOB R0,1$
10835
10836 057766 110237 057463 MOVB R2,RK5DFL
10837 057772 000207 RTS PC
    
```

```

10838
10839 ;SET UP UBEDFL
10840
10841 057774 042777 000200 124124 SIZUBE: BIC #BIT7,@UBECR1
10842 060002 032777 000200 124116 BIT #BIT7,@UBECR1
10843 060010 001403 BEQ 1$
10844 060012 112737 000001 057464 MOVB #1,UBEDFL
10845 060020 000207 1$: RTS PC
    
```

```

10846
10847 ;DETERMINE WHAT MASS BUS TESTER UNITS THERE ARE
10848
10849 060022 012737 000200 057462 SIZRH4: MOV #BIT7,RH4DFL
10850 060030 000207 RTS PC
    
```

10851  
 10852 .SBTTL DEVICE HANDLERS  
 10853 :\*\*\*\*\*

```

10854 :*
10855 :* THE FOLLOWING SIX ROUTINES:
10856 :* RH4HAN
10857 :* RP4HAN
10858 :* RS4HAN
10859 :* UBEHAN
10860 :* RK5HAN
10861 :* ARE O/I AND BUS TESTER DEVICE HANDLERS.
10862 :* THEY ARE CALLED USING:
10863 :* TRAP TABLE CALL
10864 :* FUNCTION: .BYTE
10865 :* UNITNUM: .BYTE
10866 :* DISKADR1: .WORD
10867 :* DISKADR2: .WORD
10868 :* MEMADR1: .WORD
10869 :* MEMADR2: .WORD
10870 :* WORDCNT: .WORD
10871 :* VECTOR: .WORD
10872 :* RETURN:
10873 :*A WHERE TRAP TABLE CALL IS ONE OF:
10874 :* CALRH4
10875 :* CALRP4
10876 :* CALRS4
    
```



```

10877      :*          CALUBE
10878      :*          CALRK5
10879      :*B       FUNCTION IS THE PATTERN TO BE LOADED INTO THE
10880      :*          CONTROL REGISTER FUNCTION BITS, WITH EITHER
10881      :*          INTERRUPT ENABLED OR NOT.
10882      :*C       UNITNUM IS THE DRIVE NUMBER
10883      :*D       DISKADR1 AND DISKADR2 ARE THE DISK ADDRESS
10884      :*          SECTOR NUMBER
10885      :*E       MEMADR1 AND MEMADR2 ARE THE 22-BIT MEMORY
10886      :*          ADDRESS FOR THE TRANSFER.
10887      :*F       WORDCNT IS THE WORD COUNT A POSITIVE
10888      :*          NUMBER BETWEEN 0 AND 32K.
10889      :*G       VECTOR IS THE INTERRUPT HANDLER ROUTINE SPECIFIED
10890      :*          BY THE USER FOR AN INTERRUPT ENABLED FUNCTION.
10891      :*
10892      :*          WHEN THE HANDLER PROCESSES A CALL IT RETURNS
10893      :*          WITH THE FUNCTION IN PROGRESS IF THE
10894      :*          FUNCTION WAS INTERRUPT ENABLED. WHEN THE
10895      :*          INTERRUPT OCCURS CONTROL IS GIVEN TO
10896      :*          THE USER SPECIFIED INTERRUPT HANDLER.
10897      :*          IF THE FUNCTION WAS NOT INTERRUPT
10898      :*          ENABLED THEN THE HANDLER WAITS FOR
10899      :*          FUNCTION DONE BEFORE RETURNING.
10900      :*
10901      :*          THE FLAGS:
10902      :*          XXXER1
10903      :*          XXXER2
10904      :*          XXXER3
10905      :*          WHERE XXX IS THE DEVICE, ARE USED TO
10906      :*          INDICATE AND LOG DEVICE ERRORS IN THE HANDLER.
10907      :*          XXX CAN BE RH4,RP4,RS4,UBE,RK5 OR RP3.
10908      :*          XXXER1=0          NO ERRORS
10909      :*          XXXER1=1          ERRORS WITH STATUS IN XXXER2 AND XXXER3.
10910      :*
10911      :*          *****

```

```

10912      :*          .SBTTL          RP04 DISK HANDLER
10913      :*          ;RP04 DISK HANDLER
10914      :*
10915      :*          ;REGISTERS USED IN RP4HAN
10916      :*          RP4FLG: .WORD    0
10917      :*          RP4ER1: .WORD    0          ;ERROR FLAGS.
10918      :*          RP4ER2: .WORD    0
10919      :*          RP4ER3: .WORD    0
10920      :*          RP4ER4: .WORD    0
10921      :*          RP4USE: .WORD    0
10922      :*          RP4TMP: .WORD    0
10923      :*          RP4FUN: .WORD    0
10924      :*          RP4UNI: .WORD    0
10925      :*          RP4DA1: .WORD    0
10926      :*          RP4DA2: .WORD    0
10927      :*          RP4MA1: .WORD    0
10928      :*          RP4MA2: .WORD    0
10929      :*          RP4WCT: .WORD    0
10930      :*          RP4VEC: .WORD    0
10931      :*          RP4TRK: .WORD    0
10932      :*

```

```

10933 060072 000000          RP4SEC:.WORD 0
10934 060074 000000          RP4CYL:.WORD 0
10935
10936 060076 005737 060032  RP4HAN: TST      RP4FLG      ;SEE IF THERE IS
10937 060102 001402          BEQ      RP4H1      ;ALREADY AN RP04 FUNCTION
10938 060104 104000          ERROR    ;IN PROGRESS. IF THERE
10939 060106 000000          HALT     ;IS ERROR> (SHOULD NEVER
10940 060110 012737 000340 177776 RP4H1: MOV      #340,@#PSW ;HAPPEN.)
10941 060116 011637 060046      MOV      (SP),RP4TMP ;RAISE THE PRIORITY
10942 060122 062716 000016      ADD      #16,(SP)
10943 060126 104412          SAVREG
10944 060130 013700 060046      MOV      RP4TMP,R0 ;GET AN ARGUMENT POINTER
10945 060134 112037 060050      MOV      (R0)+,RP4FUN ;RESET THE RETURN ADDRESS
10946 060140 112037 060052      MOV      (R0)+,RP4UNI ;FUNCTION
10947 060144 012037 060054      MOV      (R0)+,RP4DA1 ;UNIT, DEVICE, NUMBER
10948 060150 012037 060056      MOV      (R0)+,RP4DA2 ;DISK ADDRESS
10949 060154 012037 060060      MOV      (R0)+,RP4MA1 ;MEMORY ADDRESS
10950 060160 012037 060062      MOV      (R0)+,RP4MA2
10951 060164 012037 060064      MOV      (R0)+,RP4WCT ;WORD COUNT
10952 060170 012037 060066      MOV      (R0)+,RP4VEC ;INTERRUPT HANDLER ROUTINE
10953 060174 005037 060034      CLR      RP4ER1     ;CLEAR THE ERROR
10954 060200 005037 060036      CLR      RP4ER2     ;FLAGS
10955 060204 005037 060040      CLR      RP4ER3
10956
10957 060210 004737 060472          JSR      PC,RP4S1   ;GO SET UP THE UNIT NUMBER
10958 060214 004737 060542          JSR      PC,RP4RDY ;GET THE DEVICE READY.
10959 060220 004737 060502          JSR      PC,RP4S2   ;COMPUTE THE CYLINDER,
10960                                     ;TRACK AND SECTOR
10961 060224 004737 060526          JSR      PC,RP4S3   ;SET UP THE WORD COUNT
10962
10963 060230 013777 060052 123534 RP4H2: MOV      RP4UNI,@RP4CS2 ;SET THE RP04 REGISTERS
10964 060236 013777 060064 123520      MOV      RP4WCT,@RP4WC ;UP FOR THIS FUNCTION
10965 060244 013777 060060 123514      MOV      RP4MA1,@RP4BA
10966 060252 013777 060062 123552      MOV      RP4MA2,@RP4BAE
10967 060260 013777 060056 123502      MOV      RP4DA2,@RP4DA
10968 060266 013777 060054 123522      MOV      RP4DA1,@RP4DC
10969 060274 013700 004136      MOV      RP4V,R0    ;SET UP THE INTERRUPT
10970 060300 012720 060352      MOV      #RP4H4,(R0)+ ;VECTOR
10971 060304 012710 000340      MOV      #340,(R0)
10972 060310 013700 060050      MOV      RP4FUN,R0  ;LOAD THE FUNCTION
10973 060314 010037 060032      MOV      R0,RP4FLG ;AND GO
10974 060320 110077 123436      MOV      R0,@RP4CS1
10975 060324 032700 000100      BIT      #BIT6,R0  ;SEE IF THE FUNCTION
10976 060330 001402          BEQ      RP4H3      ;WILL INTERRUPT WHEN
10977 060332 104413          RESREG ;DONE. IF YES RETURN
10978 060334 000002          RTI     ;IF NOT INTERRUPTING
10979 060336 004737 060366      RP4H3: JSR      PC,RP4H5 ;THEN WAIT FOR THE
10980 060342 005037 060032      CLR      RP4FLG    ;FUNCTION TO FINISH.
10981 060346 104413          RESREG ;THEN RETURN.
10982 060350 000002          RTI
10983
10984 060352 005037 060032      RP4H4: CLR      RP4FLG ;WHEN THE INTERRUPT
10985 060356 004737 060366      JSR      PC,RP4H5 ;OCCURS CHECK FOR ERRORS
10986 060362 000177 177500      JMP      @RP4VEC   ;AND GO TO THE SERVICE
10987                                     ;ROUTINE.
10988

```

```

10989 060366 010046          RP4H5:  MOV    R0,-(SP)
10990 060370 053777 060052 123374  RP4H51: BIS    RP4UNI,@RP4CS2
10991 060376 017700 123360          MOV    @RP4CS1,R0
10992 060402 005700          TST    R0                ;SEE IF THE FUNCTION
10993 060404 100023          BPL    RP4H6                ;WAS COMPLETED WITHOUT
10994 060406 032700 060000          BIT    #60000,R0          ;ERRORS.
10995 060412 001420          BEQ    RP4H6
10996 060414 017737 123352 060036  MOV    @RP4CS2,RP4ER2    ;IF ERRORS OCCURRED SET
10997 060422 017737 123346 060040  MOV    @RP4DS,R1,ER3    ;THE INDICATORS
10998 060430 017737 123342 060042  MOV    @RP4RR1,RP4ER4
10999 060436 012737 177777 060034  MOV    #-1,RP4ER1
11000 060444 004737 060764          JSR    PC,RP4CLR          ;CLEAR THE CONTROL
11001 060450 012600          MOV    (SP)+,R0
11002 060452 000207          RTS    PC
11003 060454 105700          RP4H6:  TSTB   R0                ;WAIT FOR READY OR
11004 060456 100344          BPL    RP4H51              ;ERROR
11005 060460 105777 123310          TSTB   @RP4DS
11006 060464 100341          BPL    RP4H51
11007 060466 012600          MOV    (SP)+,R0
11008 060470 000207          RTS    PC
11009
11010 060472 042737 177770 060052  RP4S1:  BIC    #177770,RP4UNI    ;SET UP THE DRIVE NUMBER.
11011 060500 000207          RTS    PC
11012
11013 060502 013701 060054          RP4S2:  MOV    RP4DA1,R1      ;COMPUTE THE DISK
11014 060506 005000          CLR    R0
11015 060510 071027 000630          DIV   #408.,R0
11016 060514 010137 060054          MOV   R1,RP4DA1
11017 060520 005037 060056          CLR   RP4DA2
11018 060524 000207          RTS    PC
11019
11020 060526 005437 060064          RP4S3:  NEG    RP4WCT          ;COMPUTE VALID WORD COUNT
11021 060532 042737 177700 060062  BIC    #177700,RP4MA2    ;AND MEMORY ADDRESS
11022 060540 000207          RTS    PC
11023
11024 060542 012737 000040 060044  RP4RDY: MOV    #BIT5,RP4USE    ;CLEAR CONTROLLER AND
11025 060550 053737 060052 060044  BIS    RP4UNI,RP4USE
11026 060556 013777 060044 123206  MOV    RP4USE,@RP4CS2
11027 060564 013777 060052 123200  MOV    RP4UNI,@RP4CS2
11028 060572 105777 123164          1$:    TSTB   @RP4CS1      ;DRIVES
11029 060576 100375          BPL    1$
11030 060600 013777 060052 123164  MOV    RP4UNI,@RP4CS2
11031 060606 012777 000021 123146  MOV    #21,@RP4CS1      ;INITIALIZE THE DRIVE
11032 060614 017701 123142          2$:    MOV    @RP4CS1,R1    ;BY DOING A NOP
11033 060620 005701          TST    R1                ;WAIT FOR ERROR OR
11034 060622 100434          BMI   4$                ;READY
11035 060624 105701          TSTB   R1
11036 060626 100372          BPL    2$
11037
11038 060630 017700 123140          3$:    MOV    @RP4DS,R0      ;LOOK AT THE DRIVE
11039
11040 060634 032700 000400          BIT    #BIT8,R0          ;STATUS
11041 060640 001430          BEQ    5$                ;DRIVE PRESENT?
11042 060642 032700 000100          BIT    #BIT6,R0          ;VOLUME VALID?
11043 060646 001425          BEQ    5$
11044 060650 032700 010000          BIT    #BIT12,R0        ;ON LINE?
    
```

```

11045 060654 001422          BEQ      5$
11046 060656 032700 040000    BIT      #BIT14,R0      ;ANY ERRORS?
11047 060662 001017          BNE      5$
11048 060664 032700 004000    BIT      #BIT11,R0      ;WRITE LOCKED
11049 060670 001014          BNE      5$
11050 060672 032700 001000    BIT      #BIT9, R0      ;PROGRAMMABLE DRIVE
11051 060676 001011          BNE      5$
11052 060700 105700          TSTB    R0              ;WAIT FOR DRIVE READY
11053 060702 100344          BPL      2$
11054
11055 060704 012777 010000 123102    MOV      #BIT12,@RP4OF   ;SET 16-BIT MODE
11056 060712 000207          RTS      PC              ;RETURN READY.
11057 060714 032701 040000    4$: BIT      #BIT14,R1      ;ATTENTION OR ERROR?
11058 060720 001743          BEQ      3$
11059 060722 005726          5$: TST      (SP)+
11060 060724 017737 123042 060036    MOV      @RP4CS2,RP4ER2 ;FLAG AND RECORD
11061 060732 017737 123036 060040    MOV      @RP4DS,RP4ER3  ;ERROR
11062 060740 017737 123032 060042    MOV      @RP4RR1,RP4ER4
11063 060746 012737 177777 060034    MOV      #-1,RP4ER1
11064 060754 004737 060764    JSR      PC,RP4CLR      ;CLR THE CONTROLLER
11065 060760 104413          RESREG
11066 060762 000002          RTI
11067
11068 060764 013777 060044 123000    RP4CLR: MOV      RP4USE,@RP4CS2 ;CLR THE CONTROLLER
11069 060772 105777 122764    1$: TSTB    @RP4CS1      ;AND DRIVES.
11070 060776 100375          BPL      1$
11071 061000 000207          RTS      PC
11072

```

.SBTTL RS04 DISK HANDLE  
 ;RS04 DISK HANDLER

```

11073
11074
11075
11076 ;REGISTERS USED IN RS4HAN
11077 061002 000000    RS4FLG: .WORD 0
11078 061004 000000    RS4ER1: .WORD 0 ;ERROR FLAGS.
11079 061006 000000    RS4ER2: .WORD 0
11080 061010 000000    RS4ER3: .WORD 0
11081 061012 000000    RS4ER4: .WORD 0
11082 061014 000000    RS4USE: .WORD 0
11083 061016 000000    RS4TMP: .WORD 0
11084 061020 000000    RS4FUN: .WORD 0
11085 061022 000000    RS4UNI: .WORD 0
11086 061024 000000    RS4DA1: .WORD 0
11087 061026 000000    RS4DA2: .WORD 0
11088 061030 000000    RS4MA1: .WORD 0
11089 061032 000000    RS4MA2: .WORD 0
11090 061034 000000    RS4WCT: .WORD 0
11091 061036 000000    RS4VEC: .WORD 0
11092 061040 000000    RS4TRK: .WORD 0
11093 061042 000000    RS4SEC: .WORD 0
11094 061044 000000    RS4CYL: .WORD 0
11095
11096 061046 005737 061002    RS4HAN: TST      RS4FLG      ;SEE IF THERE ALREADY
11097 061052 001402          BEQ      RS4H1              ;IS AN RS04 FUNCTION
11098 061054 104000          ERROR
11099 061056 000000          HALT                        ;IN PROGRESS. IF SO
11100 061060 012737 000340 177776    RS4H1: MOV      #340,@#PSW   ;ERROR. (SHOULD NEVER
;HAPPEN.

```

```

11101 061066 011637 061016      MOV      (SP),RS4TMP
11102 061072 062716 000016      ADD      #16,(SP)
11103 061076 104412      SAVREG
11104 061100 013700 061016      MOV      RS4TMP,R0      ;RAISE THE PRIORITY
11105 061104 112037 061020      MOV      (R0)+,RS4FUN   ;GET A POINTER TO
11106 061110 112037 061022      MOV      (R0)+,RS4UNI   ;FUNCTION
11107 061114 012037 061024      MOV      (R0)+,RS4DA1   ;GET THE DRIVE NUMBER
11108 061120 012037 061026      MOV      (R0)+,RS4DA2   ;DISK ADDRESS
11109 061124 012037 061030      MOV      (R0)+,RS4MA1   ;MEMORY ADDRESS
11110 061130 012037 061032      MOV      (R0)+,RS4MA2
11111 061134 012037 061034      MOV      (R0)+,RS4WCT   ;WORD COUNT
11112 061140 012037 061036      MOV      (R0)+,RS4VEC   ;INTERRUPT HANDLER ADDRESS
11113 061144 005037 061004      CLR      RS4ER1         ;CLEAR THE ERROR FLAGS
11114 061150 005037 061006      CLR      RS4ER2
11115 061154 005037 061010      CLR      RS4ER3
11116
11117 061160 004737 061434      JSR      PC,RS4S1       ;SET UP UNIT (DRIVE) NUMBER
11118 061164 004737 061526      JSR      PC,RS4RDY     ;INITIALIZE DRIVE AND
11119                                     ;CONTROLLER
11120 061170 004737 061444      JSR      PC,RS4S2     ;COMPUTE TRACK AND SECTOR
11121 061174 004737 061512      JSR      PC,RS4S3     ;COMPUTE WORD COUNT.
11122
11123 061200 013777 061022 122526 RS4H2: MOV      RS4UNI,@RS4CS2 ;SET UP THE CONTROL
11124 061206 013777 061034 122512      MOV      RS4WCT,@RS4WC ;AND DRIVE REGISTERS
11125 061214 013777 061030 122506      MOV      RS4MA1,@RS4BA
11126 061222 013777 061032 122524      MOV      RS4MA2,@RS4BAE
11127 061230 013777 061024 122474      MOV      RS4DA1,@RS4DA
11128 061236 013700 004134      MOV      RS4V,R0
11129 061242 012720 061314      MOV      #RS4H4,(R0)+ ;SET THE INTERRUPT
11130 061246 012710 000340      MOV      #340,(R0)
11131 061252 013700 061020      MOV      RS4FUN,R0
11132 061256 010037 061002      MOV      R0,RS4FLG
11133 061262 110077 122436      MOV      R0,@RS4CS1   ;LOAD THE FUNCTION AND GO.
11134 061266 032700 000100      MOV      R0,@RS4CS1   ;SEE IF AN INTERRUPT
11135 061272 001402      BIT      #BIT6,R0     ;IS TO BE EXPECTED.
11136 061274 104413      BEQ     RS4H3        ;IF YES THEN RETURN
11137 061276 000002      RESREG
11138                                     RTI
11139 061300 004737 061330      RS4H3: JSR      PC,RS4H5   ;IF NOT INTERRUPTING
11140 061304 005037 061002      CLR      RS4FLG      ;THEN WAIT FOR THE
11141 061310 104413      RESREG              ;FUNCTION TO FINISH
11142 061312 000002      RTI
11143
11144 061314 005037 061002      RS4H4: CLR      RS4FLG ;WHEN THE INTERRUPT OCCURS.
11145 061320 004737 061330      JSR      PC,RS4H5   ;MAKE SURE THERE WERE
11146 061324 000177 177506      JMP     @RS4VEC     ;NO ERRORS BEFORE GOING
11147                                     ;TO THE INTERRUPT
11148                                     ;SERVICE ROUTINE.
11149 061330 010046      RS4H5: MOV      R0,-(SP)
11150 061332 053777 061022 122374 RS4H51: BIS      RS4UNI,@RS4CS2
11151 061340 017700 122360      MOV      @RS4CS1,R0
11152 061344 005700      TST     R0          ;SEE IF THE FUNCTION
11153 061346 100023      BPL     RS4H6       ;WAS COMPLETED WITHOUT
11154 061350 032700 060000      BIT      #60000,R0   ;ERRORS
11155 061354 001420      BEQ     RS4H6
11156 061356 017737 122352 061006      MOV      @RS4CS2,RS4ER2 ;IF ERRORS OCCURRED
    
```

```

11157 061364 017737 122346 061010      MOV      @RS4DS,RS4ER3      ;SET THE INDICATORS
11158 061372 017737 122342 061012      MOV      @RS4ER,RS4ER4
11159 061400 012737 177777 061004      MOV      #-1,RS4ER1
11160 061406 004737 061720      JSR      PC,RS4CLR        ;THEN CLEAR THE CONTROL
11161 061412 012600      MOV      (SP)+,R0
11162 061414 000207      RTS      PC                ;AND DRIVES
11163 061416 105700      RS4H6:  TSTB     R0
11164 061420 100344      BPL     RS4H51             ;WAIT FOR READY OR
11165 061422 105777 122310      TSTB     @RS4DS           ;ERROR
11166 061426 100341      BPL     RS4H51
11167 061430 012600      MOV      (SP)+,R0
11168 061432 000207      RTS      PC
11169
11170 061434 042737 177770 061022  RS4S1:  BIC      #177770,RS4UNI  ;SET UP DRIVE NUMBER
11171 061442 000207      RTS      PC
11172
11173 061444 013701 061024      RS4S2:  MOV      RS4DA1,R1      ;COMPUTE A DISK
11174 061450 005000      CLR     R0                ;ADDRESS
11175 061452 071027 007000      DIV     #3584.,R0
11176 061456 005000      CLR     R0
11177 061460 071027 000100      DIV     #100,R0
11178 061464 010037 061040      MOV     R0,RS4TRK
11179 061470 010137 061044      MOV     R1,RS4CYL
11180 061474 000300      SWAB   R0
11181 061476 006200      ASR    R0
11182 061500 006200      ASR    R0
11183 061502 050001      BIS    R0,R1
11184 061504 010137 061024      MOV     R1,RS4DA1
11185 061510 000207      RTS      PC
11186
11187 061512 005437 061034      RS4S3:  NEG     RS4WCT          ;COMPUTE A VALID WORD
11188 061516 042737 177700 061032  BIC     #177700,RS4MA2    ;COUNT AND MEMORY
11189 061524 000207      RTS      PC                ;ADDRESS
11190 061526 012737 000040 061014  RS4RDY: MOV     #BIT5,RS4USE    ;CLEAR CONTROLLER AND DRIVES
11191 061534 053737 061022 061014  BIS     RS4UNI,RS4USE
11192 061542 013777 061014 122164  MOV     RS4USE,@RS4CS2
11193 061550 013777 061022 122156  MOV     RS4UNI,@RS4CS2
11194 061556 105777 122142      1$:    TSTB     @RS4CS1
11195 061562 100375      BPL     1$
11196 061564 013777 061022 122142  MOV     RS4UNI,@RS4CS2
11197 061572 012777 000001 122124  MOV     #1,@RS4CS1        ;INITIALIZE THE DRIVE
11198 061600 017701 122120      2$:    MOV     @RS4CS1,R1      ;BY DOING A NOP.
11199 061604 005701      TST     R1
11200 061606 100420      BMI     4$
11201 061610 105701      TSTB   R1
11202 061612 100372      BPL     2$
11203
11204 061614 017700 122116      3$:    MOV     @RS4DS,R0        ;LOOK AT THE DRIVE STATUS
11205 061620 032700 000400      BIT     #BIT8,R0          ;DRIVE PRESENT?
11206 061624 001414      BEQ     5$
11207 061626 032700 010000      BIT     #BIT12,R0        ;ON LINE?
11208 061632 001411      BEQ     5$
11209 061634 032700 004000      BIT     #BIT11,R0        ;WRITE LOCKED?
11210 061640 001006      BNE     5$
11211 061642 105700      TSTB   R0                ;DRIVE READY?
11212 061644 100355      BPL     2$
    
```

```

11213 061646 000207          RTS      PC
11214 061650 032701 040000    4$:     BIT      #BIT14,R1      ;ATTENTION OR ERROR?
11215 061654 001757          BEQ      3$
11216 061656 005726          5$:     TST      (SP)+
11217 061660 017737 122050 061006    MOV      @RS4CS2,RS4ER2    ;FLAG AND RECORD THE
11218 061666 017737 122044 061010    MOV      @RS4DS,RS4ER3    ;ERROR
11219 061674 017737 122040 061012    MOV      @RS4ER,RS4ER4
11220 061702 012737 177777 061004    MOV      #-1,RS4ER1
11221 061710 004737 061720    JSR      PC,RS4CLR        ;CLR THE CONTROLLER
11222 061714 104413          RESREG
11223 061716 000002          RTI          ;AND DRIVES AND RETURN.
11224
11225 061720 013777 061014 121776    RS4CLR: MOV      RS4USE,@RS4CS1 ;CLR THE CONTROLLER
11226 061726 105777 121772    1$:     TSTB     @RS4CS1
11227 061732 100375          BPL      1$
11228 061734 000207          RTS      PC
11229
11230
11231
11232
11233
11234

```

.SBTTL RK05 DISK HANDLER  
 ;RK05 DISK HANDLER

;REGISTERS USED IN RK5HAN

```

11235 061736 000000    RK5FLG: .WORD 0
11236 061740 000000    RK5ER1: .WORD 0      ;ERROR FLAGS.
11237 061742 000000    RK5ER2: .WORD 0
11238 061744 000000    RK5ER3: .WORD 0
11239 061746 000000    RK5ER4: .WORD 0
11240 061750 000000    RK5USE: .WORD 0
11241 061752 000000    RK5TMP: .WORD 0
11242 061754 000000    RK5FUN: .WORD 0
11243 061756 000000    RK5UNI: .WORD 0
11244 061760 000000    RK5DA1: .WORD 0
11245 061762 000000    RK5DA2: .WORD 0
11246 061764 000000    RK5MA1: .WORD 0
11247 061766 000000    RK5MA2: .WORD 0
11248 061770 000000    RK5WCT: .WORD 0
11249 061772 000000    RK5VEC: .WORD 0
11250 061774 000000    RK5TRK: .WORD 0
11251 061776 000000    RK5SEC: .WORD 0
11252 062000 000000    RK5CYL: .WORD 0
11253

```

```

11254 062002 005737 061736    RK5HAN: TST      RK5FLG      ;SEE IF THERE IS ALREADY AN
11255 062006 001402          BEQ      RK5H1      ;RK05 FUNCTION IN PROGRESS
11256 062010 104000          ERROR
11257 062012 000000          HALT
11258

```

```

11259 062014 012737 000340 177776    RK5H1:  MOV      #340,@#PSW    ;RAISE THE PRIORITY
11260 062022 011637 061752    MOV      (SP),RK5TMP
11261 062026 062716 000016    ADD      #16,(SP)
11262 062032 104412          SAVREG
11263 062034 013700 061752    MOV      RK5TMP,R0
11264 062040 112037 061754    MOV      (R0)+,RK5FUN    ;GET THE ARGUMENTS.
11265 062044 112037 061756    MOV      (R0)+,RK5UNI
11266 062050 012037 061760    MOV      (R0)+,RK5DA1
11267 062054 012037 061762    MOV      (R0)+,RK5DA2
11268 062060 012037 061764    MOV      (R0)+,RK5MA1

```

```

11269 062064 012037 061766      MOV      (R0)+,RK5MA2
11270 062070 012037 061770      MOV      (R0)+,RK5WCT
11271 062074 012037 061772      MOV      (R0)+,RK5VEC
11272
11273 062100 005037 061740      CLR      RK5ER1      ;CLR THE ERROR FLAGS
11274 062104 005037 061742      CLR      RK5ER2
11275 062110 005037 061744      CLR      RK5ER3
11276
11277 062114 004737 062364      JSR      PC,RK5S1      ;SET UP THE DRIVE NUMBER
11278 062120 004737 062570      JSR      PC,RK5RDY     ;GET THE DEVICE AND CONTROL
11279                                     ;READY
11280 062124 004737 062406      JSR      PC,RK5S2     ;COMPUTE THE SURFACE
11281                                     ;CYLINDER AND SECTOR
11282                                     ;ADDRESS.
11283 062130 004737 062510      JSR      PC,RK5S3     ;SET UP A WORD COUNT,
11284                                     ;THE UNIBUS MAP
11285                                     ;AND BUS ADDRESS.
11286
11287 062134 005077 121744      RK5H2: CLR      @RK5CS1
11288 062140 013777 061756      MOV      RK5UNI,@RK5DA ;SET THE DEVICE REGISTERS
11289 062146 013777 061770      MOV      RK5WCT,@RK5WC ;TO DO THE FUNCTION
11290 062154 013777 061764      MOV      RK5MA1,@RK5BA
11291 062162 053777 061766      BIS      RK5MA2,@RK5CS1
11292 062170 053777 061760      BIS      RK5DA1,@RK5DA
11293 062176 013700 004142      MOV      RK5V,R0      ;LOAD THE INTERRUPT VECTOR
11294 062202 012720 062254      MOV      #RK5H4,(R0)+
11295 062206 012710 000340      MOV      #340,(R0)
11296 062212 013700 061754      MOV      RK5FUN,R0
11297 062216 010037 061736      MOV      R0,RK5FLG
11298 062222 050077 121656      BIS      R0,@RK5CS1   ;LOAD THE FUNCTION AND
11299                                     ;GO
11300
11301 062226 032700 000100      BIT      #BIT6,R0     ;SEE IF THE FUNCTION WILL
11302 062232 001402                                     BEQ      RK5H3        ;INTERRUPT WHEN DONE.
11303 062234 104413                                     RESREG
11304 062236 000002                                     RTI          ;IF YES RETURN
11305
11306 062240 004737 062302      RK5H3: JSR      PC,RK5H5   ;IF THE FUNCTION WAS
11307 062244 005037 061736      CLR      RK5FLG      ;NOT INTERRUPT ENABLED
11308 062250 104413                                     RESREG
11309 062252 000002                                     RTI          ;WAIT FOR DONE OR ERROR.
11310
11311 062254 004737 062302      RK5H4: JSR      PC,RK5H5   ;SEE IF THERE WERE ANY ERRORS.
11312 062260 005037 061736      CLR      RK5FLG
11313 062264 012777 062300      MOV      #1$,@RK5V
11314 062272 000230                                     SPL      0
11315 062274 000177 177472      JMP      @RK5VEC
11316 062300 000002      1$:      RTI
11317
11318 062302 010046      RK5H5: MOV      R0,-(SP)
11319 062304 017700 121574      RK5H51: MOV      @RK5CS1,R0 ;SEE IF ANY ERROR OCCURRED
11320 062310 005700                                     TST      R0
11321 062312 100015                                     BPL      RK5H6
11322 062314 017737 121562      MOV      @RK5ER,RK5ER2 ;IF YES, FLAG THE ERROR
11323 062322 017737 121552      MOV      @RK5DS,RK5ER3 ;AND SAVE THE STATUS
11324 062330 012737 177777      MOV      #-1,RK5ER1
    
```





```

11381 062604 105777 121274      1$:  TSTB  @RK5CS1
11382 062610 100375                BPL  1$
11383
11384 062612 053777 061756 121272      BIS  RK5UNI,@RK5DA  ;DO A DRIVE CLEAR
11385 062620 012777 000015 121256      MOV  #15,@RK5CS1  ;FUNCTION
11386
11387 062626 017701 121252      2$:  MOV  @RK5CS1,R1  ;WAIT FOR DONE OR
11388 062632 100420                BMI  5$           ;ERROR.
11389 062634 105701                TSTB R1
11390 062636 100373                BPL  2$
11391
11392 062640 017701 121234      3$:  MOV  @RK5DS,R1
11393 062644 032701 000040                BIT  #BIT5,R1  ;WRITE ENABLED?
11394 062650 001011                BNE  5$
11395 062652 005777 121224                TST  @RK5ER
11396 062656 100406                BMI  5$
11397 062660 105701                TSTB R1
11398 062662 100366                BPL  3$
11399 062664 032701 000100                BIT  #BIT6,R1
11400 062670 001763                BEQ  3$
11401 062672 000207      4$:  RTS  PC
11402
11403 062674 005726      5$:  TST  (SP)+
11404 062676 017737 121200 061742      MOV  @RK5ER,RK5ER2
11405 062704 017737 121170 061744      MOV  @RK5DS,RK5ER3
11406 062712 012737 177777 061740      MOV  #-1,RK5ER1
11407 062720 004737 062730                JSR  PC,RK5CLR
11408 062724 104413                RESREG
11409 062726 000002                RTI
11410
11411 062730 005077 121156      RK5CLR: CLR  @RK5DA  ;RESET THE CONTROLLER
11412 062734 012777 000001 121142      MOV  #1,@RK5CS1  ;BY DOING A CONTROL
11413 062742 105777 121136      1$:  TSTB  @RK5CS1  ;CLEAR FUNCTION
11414 062746 100375                BPL  1$
11415 062750 000207                RTS  PC
    
```

.SBTTL UNIBUS EXERCISER HANDLER  
 ;UNIBUS EXERCISER HANDLER

```

11416
11417
11418
11419
11420      ;REGISTERS USED IN UBEHAN
11421 062752 000000      UBEFLG: .WORD 0
11422 062754 000000      UBEER1: .WORD 0      ;ERROR FLAGS.
11423 062756 000000      UBEER2: .WORD 0
11424 062760 000000      UBEER3: .WORD 0
11425 062762 000000      UBEER4: .WORD 0
11426 062764 000000      UBEUSE: .WORD 0
11427 062766 000000      UBETMP: .WORD 0
11428 062770 000000      UBEFUN: .WORD 0
11429 062772 000000      UBEUNI: .WORD 0
11430 062774 000000      UBEDA1: .WORD 0
11431 062776 000000      UBEDA2: .WORD 0
11432 063000 000000      UBEMA1: .WORD 0
11433 063002 000000      UBEMA2: .WORD 0
11434 063004 000000      UBEWCT: .WORD 0
11435 063006 000000      UBEVEC: .WORD 0
11436 063010 000000      UBETRK: .WORD 0
    
```

```

11437 063012 000000          UBESec:.WORD 0
11438 063014 000000          UBECYL:.WORD 0
11439
11440 063016 005737 062752    UBEHAN: TST      UBFLG      ;SEE IF THERE IS ALREADY
11441 063022 001402          BEQ      UBEH1      ;A UNIBUS EXERCISER FUNCTION
11442 063024 104000          ERROR    ;IN PROGRESS. IF THERE
11443 063026 000000          HALT     ;IS ERROR. (SHOULD NEVER HAPPEN)
11444
11445 063030 012737 000340 177776 UBEH1: MOV      #340,@#PSW    ;RAISE THE PRIORITY
11446 063036 011637 062766      MOV      (SP),UBETMP ;GET AN ARGUMENT POINTER
11447 063042 062716 000016      ADD      #16,(SP)
11448 063046 104412          SAVREG
11449 063050 013700 062766      MOV      UBETMP,RO    ;RESET THE RETURN ADDRESS
11450
11451 063054 012037 062770      MOV      (RO)+,UBEFUN ;GET THE ARGUMENTS.
11452 063060 012037 062774      MOV      (RO)+,UBEDA1
11453 063064 012037 062776      MOV      (RO)+,UBEDA2
11454 063070 012037 063000      MOV      (RO)+,UBEMA1
11455 063074 012037 063002      MOV      (RO)+,UBEMA2
11456 063100 012037 063004      MOV      (RO)+,UBEWCT
11457 063104 012037 063006      MOV      (RO)+,UBEVEC
11458 063110 005037 062754          CLR      UBEER1      ;CLEAR THE ERROR FLAGS
11459 063114 005037 062756          CLR      UBEER2
11460 063120 005037 062760          CLR      UBEER3
11461 063124 004737 063406          JSR      PC,UBERDY
11462 063130 004737 063332          JSR      PC,UBES1    ;GO SET UP THE BUS
11463                                     ;ADDRESS AND UB MAP
11464
11465 063134 013777 063004 120760 UBEH2: MOV      UBEWCT,@UBECC ;SET THE DEVICE
11466 063142 012777 060000 120754      MOV      #60000,@UBEDA ;REGISTERS
11467 063150 053777 063002 120754      BIS      UBEMA2,@UBECR2
11468 063156 013777 062776 120734      MOV      UBEDA2,@UBEDB
11469 063164 013700 004144          MOV      UBEV,RO
11470 063170 012720 063242          MOV      #UBEH4,(RO)+
11471 063174 012710 000340          MOV      #340,(RO)
11472 063200 013700 062770          MOV      UBEFUN,RO
11473 063204 010037 062752          MOV      RO,UBEFLG
11474 063210 010077 120712          MOV      RO,@UBECR1    ;LOAD THE FUNCTION
11475 063214 032700 000100          BIT      #BIT6,RO      ;SEE IF THE FUNCTION
11476 063220 001402          BEQ      UBEH3      ;IS INTERRUPT ENABLED
11477 063222 104413          RESREG ;IF YES RETURN
11478 063224 000002          RTI
11479
11480 063226 004737 063256          UBEH3: JSR      PC,UBEH5 ;IF NOT INTERRUPT ENABLED
11481 063232 005037 062752          CLR      UBFLG      ;WAIT FOR DONE OR
11482 063236 104413          RESREG ;ERROR
11483 063240 000002          RTI
11484
11485 063242 005037 062752          UBEH4: CLR      UBFLG      ;WHEN THE INTERRUPT
11486 063246 004737 063256          JSR      PC,UBEH5 ;OCCURS SEE IF ANY ERRORS
11487 063252 000177 177530          JMP      @UBEVEC ;OCCURRED
11488
11489 063256 010046          UBEH5: MOV      RO,-(SP)
11490 063260 017700 120642          UBEH51: MOV      @UBECR1,RO ;WAIT FOR DONE OR
11491 063264 005700          TST      RO ;ERROR
11492 063266 100015          BPL      UBEH6

```

```

11493
11494 063270 017737 120632 062756      MOV      @UBECR1,UBEER2
11495 063276 017737 120630 062760      MOV      @UBECR2,UBEER3
11496 063304 012737 177777 062754      MOV      #-1,UBEER1
11497 063312 004737 063462      JSR      PC,UBCLR
11498 063316 012600      MOV      (SP)+,R0
11499 063320 000207      RTS      PC
11500
11501 063322 105700      UBEH6:  TSTB   R0
11502 063324 100355      BPL     UBEH51
11503 063326 012600      MOV     (SP)+,R0
11504 063330 000207      RTS     PC
11505
11506 063332 013700 063000      UBES1:  MOV     UBEMA1,R0      ;SET UP THE BUS ADDRESS
11507 063336 013701 063002      MOV     UBEMA2,R1      ;AND UB MAPPING BOX
11508 063342 042701 177700      BIC     #177700,R1
11509 063346 012702 170214      MOV     #MAPLO3,R2
11510
11511 063352 010022      1$:    MOV     R0,(R2)+
11512 063354 010122      MOV     R1,(R2)+
11513 063356 062700 020000      ADD     #20000,R0
11514 063362 005501      ADC     R1
11515
11516 063364 005037 063002      CLR     UBEMA2
11517 063370 005037 063000      CLR     UBEMA1
11518 063374 005137 063004      COM     UBEWCT
11519 063400 005237 063004      INC     UBEWCT
11520 063404 000207      RTS     PC
11521
11522 063406 005077 120516      UBERDY: CLR     @UBECLR      ;TRY TO GET DEVICE
11523                                     ;READY
11524 063412 017700 120510      1$:    MOV     @UBECR1,R0
11525 063416 100403      BMI     2$
11526 063420 105700      TSTB   R0
11527 063422 100373      BPL     1$
11528 063424 000207      RTS     PC
11529
11530 063426 005726      2$:    TST     (SP)+
11531 063430 017737 120472 062756      MOV     @UBECR1,UBEER2
11532 063436 017737 120470 062760      MOV     @UBECR2,UBEER3
11533 063444 012737 177777 062760      MOV     #-1,UBEER3
11534 063452 004737 063462      JSR     PC,UBCLR
11535 063456 104413      RESREG
11536 063460 000002      RTI
11537
11538 063462 005077 120442      UBCLR:  CLR     @UBECLR      ;CLEAR THE DEVICE.
11539 063466 105777 120434      1$:    TSTB   @UBECR1
11540 063472 100375      BPL     1$
11541 063474 000207      RTS     PC
11542
11543                                     .SBTTL      MASS BUS TESTER HANDLER
11544                                     ;THIS CODE IS FOR HANDLING THE MASS BUS
11545                                     ;TESTED DEVICE.
11546
11547                                     ;REGISTERS USED IN RH4HAN
11548 063476 000000      RH4FLG: .WORD  0
    
```

```

11549 063500 000000 RH4ER1: .WORD 0 ;ERROR FLAGS.
11550 063502 000000 RH4ER2: .WORD 0
11551 063504 000000 RH4ER3: .WORD 0
11552 063506 000000 RH4ER4: .WORD 0
11553 063510 000000 RH4USE: .WORD 0
11554 063512 000000 RH4TMP: .WORD 0
11555 063514 000000 RH4FUN: .WORD 0
11556 063516 000000 RH4UNI: .WORD 0
11557 063520 000000 RH4DA1: .WORD 0
11558 063522 000000 RH4DA2: .WORD 0
11559 063524 000000 RH4MA1: .WORD 0
11560 063526 000000 RH4MA2: .WORD 0
11561 063530 000000 RH4WCT: .WORD 0
11562 063532 000000 RH4VEC: .WORD 0
11563 063534 000000 RH4TRK: .WORD 0
11564 063536 000000 RH4SEC: .WORD 0
11565 063540 000000 RH4CYL: .WORD 0
11566
11567 063542 005737 063476 RH4HAN: TST RH4FLG ;SEE IF A FUNCTION
11568 063546 001402 BEQ RH4H1 ;IS ALREADY ACTIVE IF
11569 063550 104000 ERROR ;SO ERROR.
11570 063552 000000 HALT
11571
11572 063554 012777 000340 114214 RH4H1: MOV #340,@PSW ;RAISE THE PRIORITY
11573 063562 011637 063512 MOV (SP),RH4TMP
11574 063566 062716 000016 ADD #16,(SP)
11575 063572 104412 SAVREG
11576 063574 013700 063512 MOV RH4TMP,R0 ;RESET THE RETURN
11577 063600 112037 063514 MOVB (R0)+,RH4FUN
11578 063604 112037 063516 MOVB (R0)+,RH4UNI
11579 063610 012037 063520 MOV (R0)+,RH4DA1
11580 063614 012037 063522 MOV (R0)+,RH4DA2
11581 063620 012037 063524 MOV (R0)+,RH4MA1
11582 063624 012037 063526 MOV (R0)+,RH4MA2
11583 063630 012037 063530 MOV (R0)+,RH4WCT
11584 063634 011037 063532 MOV (R0),RH4VEC
11585 063640 005037 063500 CLR RH4ER1 ;CLEAR THE ERROR FLAGS
11586 063644 005037 063502 CLR RH4ER2
11587 063650 005037 063504 CLR RH4ER3
11588 063654 004737 064134 JSR PC,RH4S1 ;SET UP THE UNIT NUMBER
11589 063660 004737 064160 JSR PC,RH4RDY ;GET THE UNIT READY
11590 063664 004737 064144 JSR PC,RH4S2
11591
11592 063670 013777 063516 120152 RH4H2: MOV RH4UNI,@RH4CS2 ;SET THE CONTROL REGISTERS
11593 063676 013777 063530 120136 MOV RH4WCT,@RH4WC ;AND DEVICE REGISTERS
11594 063704 013777 063524 120132 MOV RH4MA1,@RH4BA
11595 063712 013777 063526 120152 MOV RH4MA2,@RH4AE
11596 063720 013777 063520 120132 MOV RH4DA1,@RH4DR
11597 063726 012777 004000 120130 MOV #4000,@RH4MR1
11598 063734 000240 NOP
11599 063736 013700 004140 MOV RH4V,R0 ;VECTOR
11600 063742 012720 064014 MOV #RH4H4,(R0)+
11601 063746 012710 000340 MOV #340,(R0)
11602 063752 013700 063514 MOV RH4FUN,R0
11603 063756 010C37 063476 MOV R0,RH4FLG ;LOAD THE FUNCTION AND
11604 063762 110077 120052 MOVB R0,@RH4CS1 ;GO
    
```

```

11605 063766 032700 000100          BIT      #BIT6,R0      ;SEE IF THIS FUNCTION
11606 063772 001402          BEQ      RH4H3        ;WILL INTERRUPT WHEN DONE
11607 063774 104413          RESREG   RTI         ;IF YES RETURN TO CALL
11608 063776 000002
11609
11610 064000 004737 064030          RH4H3: JSR      PC,RH4H5 ;IF NOT INTERRUPT
11611 064004 005037 063476          CLR      RH4FLG      ;ENABLED WAIT FOR
11612 064010 104413          RESREG   RTI         ;THE FUNCTION TO
11613 064012 000002          RTI         ;FINISH THEN RETURN.
11614
11615 064014 005037 063476          RH4H4: CLR      RH4FLG ;WHEN THE INTERRUPT
11616 064020 004737 064030          JSR      PC,RH4H5   ;OCCURS CHECKS FOR
11617 064024 000177 177502          JMP      @RH4VEC    ;ERRORS. THEN GO TO THE
11618                                     ;SPECIFIED SERVICE
11619                                     ;ROUTINE
11620
11621 064030 010046          RH4H5: MOV      R0,-(SP)
11622 064032 053777 063516 120010 RH4H51: BIS      RH4UNI,@RH4CS2
11623 064040 017700 117774          MOV      @RH4CS1,R0 ;SEE IF THE FUNCTION
11624 064044 005700          TST      R0         ;WAS COMPLETED WITHOUT
11625 064046 100023          BPL      RH4H6      ;ERRORS.
11626 064050 032700 060000          BIT      #60000,R0
11627 064054 001420          BEQ      RH4H6
11628 064056 017737 117766 063502          MOV      @RH4CS2,RH4ER2 ;IF ERRORS OCCURRED
11629 064064 017737 117762 063504          MOV      @RH4ST,RH4ER3 ;SAVE STATUS AND SET
11630 064072 017737 117756 063506          MOV      @RH4ER,RH4ER4
11631 064100 012737 177777 063500          MOV      #-1,RH4ER1 ;ERROR FLAGS.
11632 064106 004737 064352          JSR      PC,RH4CLR
11633 064112 012600          MOV      (SP)+,R0
11634 064114 000207          RTS      PC
11635
11636 064116 105700          RH4H6: TSTB    R0      ;WAIT FOR READY OR
11637 064120 100344          BPL      RH4H51     ;ERROR
11638 064122 105777 117724          TSTB    @RH4ST
11639 064126 100341          BPL      RH4H51
11640 064130 012600          MOV      (SP)+,R0
11641 064132 000207          RTS      PC
11642
11643 064134 042737 177770 063516 RH4S1: BIC      #177770,RH4UNI ;SET UP THE DRIVE NUMBER
11644 064142 000207          RTS      PC
11645
11646 064144 012737 000000 063522 RH4S2: MOV      #0,RH4DA2 ;FOR DEBUG.
11647 064152 005437 063530          NEG      RH4WCT     ;SET UP WORD COUNT
11648 064156 000207          RTS      PC
11649
11650 064160 012737 000040 063510 RH4RDY: MOV      #BIT5,RH4USE ;CLR THE CONTROLLER
11651 064166 053737 063516 063510          BIS      RH4UNI,RH4USE
11652 064174 013777 063510 117646          MOV      RH4USE,@RH4CS2
11653 064202 013777 063516 117640          MOV      RH4UNI,@RH4CS2
11654 064210 105777 117624          1$: TSTB    @RH4CS1 ;AND DRIVES
11655 064214 100375          BPL      1$
11656 064216 013777 063516 117624          MOV      RH4UNI,@RH4CS2 ;DO A NOP FUNCTION
11657 064224 012777 000001 117606          MOV      #1,@RH4CS1 ;TO INITIALIZE THE
11658                                     ;DRIVE
11659 064232 017701 117602          2$: MOV      @RH4CS1,R1 ;WAIT FOR READY OR ERROR.
11660 064236 005701          TST      R1

```

```

11661 064240 100420          BMI      4$
11662 064242 105701          TSTB    R1
11663 064244 100372          BPL     2$
11664
11665 064246 017700 117600      3$:    MOV    @RH4ST,R0      ;LOOK AT THE UNIT STATUS
11666 064252 032700 000400          BIT    #BIT8,R0      ;UNIT PRESENT?
11667 064256 001414          BEQ    5$
11668 064260 032700 010000          BIT    #BIT12,R0     ;ON LINE?
11669 064264 001411          BEQ    5$
11670 064266 032700 040000          BIT    #BIT14,R0     ;ANY ERRORS?
11671 064272 001006          BNE    5$
11672 064274 105700          TSTB   R0            ;WAIT FOR UNIT READY
11673 064276 100355          BPL    2$
11674 064300 000207          RTS    PC
11675
11676 064302 032701 040000      4$:    BIT    #BIT14,R1     ;ATTENTION OR ERROR
11677 064306 001757          BEQ    3$
11678 064310 005726          5$:    TST    (SP)+        ;FLAG AND RECORD ERROR
11679 064312 017737 117532 063502          MOV    @RH4CS2,RH4ER2
11680 064320 017737 117526 063504          MOV    @RH4ST,RH4ER3
11681 064326 017737 117522 063506          MOV    @RH4ER,RH4ER4
11682 064334 012737 177777 063500          MOV    #-1,RH4ER1
11683 064342 004737 064352          JSR    PC,RH4CLR
11684 064346 104413          RESREG
11685 064350 000002          RTI
11686
11687 064352 013777 063510 117470  RH4CLR: MOV    RH4USE,@RH4CS2 ;CLR THE CONTROLLER
11688 064360 105777 117454      1$:    TSTB   @RH4CS1     ;AND DRIVES.
11689 064364 100375          BPL    1$
11690 064366 000207          TSTD1: RTS    PC
11691 064370 001000          .BLKW  512.
11692          ;SPECIAL MESSAGES:
11693
11694 066370 041536 000200          CONCMS: .ASCIZ  '^C'<CRLF>
11695
11696 066374 047515 044516 047524  MMESRS: .ASCIZ  'MONITOR (OR LOADER) RESTORED!'<CRLF>
11697 066402 020122 047450 020122
11698 066410 047514 042101 051105
11699 066416 020051 042522 052123
11700 066424 051117 042105 100041
11701 066432      000
11702
11703 066433      200 047520 042527  POWERM: .ASCIZ  <CRLF>'POWER FAILURE, PROGRAM RESTARTING'<CRLF><CRLF>
11704 066440 020122 040506 046111
11705 066446 051125 026105 050040
11706 066454 047522 051107 046501
11707 066462 051040 051505 040524
11708 066470 052122 047111 100107
11709 066476 000200
11710
11711 066500 000011          $TAB:   .ASCIZ  <TAB>
11712
11713 066502 042600 050130 041505  MTA5:   .ASCII  <CRLF>'EXPECTED DATA:'<CRLF>
11714 066510 042524 020104 040504
11715 066516 040524 100072
11716 066522 051107 052517 020120          .ASCIZ  'GROUP 0.GROUP 1.MEM EV.'<TAB>'MEM ODD.'<CRLF>

```

11717	066530	027060	051107	052517	
11718	066536	020120	027061	042515	
11719	066544	020115	053105	004456	
11720	066552	042515	020115	042117	
11721	066560	027104	000200		
11722					
11723	066564	042200	052101	020101	MTA11: .ASCII <CRLF>'DATA WRITTEN.'<TAB>'TEST ADDR.'<TAB>'ERROR REG.'<CRLF>
11724	066572	051127	052111	042524	
11725	066600	027116	052011	051505	
11726	066606	020124	042101	051104	
11727	066614	004456	051105	047522	
11728	066622	020122	042522	027107	
11729	066630	200			
11730					
11731	066631	040	047111	000040	MTA17: .ASCIZ ' IN '
11732					
11733	066636	054105	042520	052103	MTB17: .ASCIZ 'EXPECTED DATA:'<CRLF>
11734	066644	042105	042040	052101	
11735	066652	035101	000200		
11736					
11737	066656	054502	042524	004456	MTC17: .ASCIZ 'BYTE.'<TAB>
11738	066664	000			
11739					
11740	066665	127	051117	027104	MTA20: .ASCIZ 'WORD.'<TAB>
11741	066672	000011			
11742					
11743	066674	054105	042520	052103	MTA21: .ASCII 'EXPECTED DATA:'<CRLF>
11744	066702	042105	042040	052101	
11745	066710	035101	200		
11746	066713	110	052111	020123	.ASCIZ 'HITS IN GROUP 0.'<TAB>'/'<TAB>'HITS IN GROUP 1. '<CRLF>
11747	066720	047111	043440	047522	
11748	066726	050125	030040	004456	
11749	066734	004457	044510	051524	
11750	066742	044440	020116	051107	
11751	066750	052517	020120	027061	
11752	066756	100040	000		
11753					
11754		066631			MTB21=MTA17
11755					
11756	066761	200	042524	052123	MTA43: .ASCII <CRLF>'TEST ADDRESS.'<TAB>'ERROR ADRS REG.'<TAB>
11757	066766	040440	042104	042522	
11758	066774	051523	004456	051105	
11759	067002	047522	020122	042101	
11760	067010	051522	051040	043505	
11761	067016	004456			
11762	067020	051105	047522	020122	.ASCIZ 'ERROR REG.'<CRLF>
11763	067026	042522	027107	000200	
11764					
11765	067034	053600	047522	042524	MTA45: .ASCIZ <CRLF>'WROTE. 377'<TAB>'IN BYTE. '
11766	067042	020056	033463	004467	
11767	067050	047111	041040	052131	
11768	067056	027105	000040		
11769					
11770	067062	051200	040505	020104	MTB45: .ASCIZ <CRLF>'READ DATA. '
11771	067070	040504	040524	020056	
11772	067076	000			



```

11773
11774 067077 011 047111 053440 MTC45: .ASCIZ <TAB>'IN WORD. '
11775 067104 051117 027104 000040
11776
11777 067112 053600 047522 042524 MTA50: .ASCIZ <CRLF>'WROTE. 000'<TAB>'IN BYTE. '
11778 067120 020056 030060 004460
11779 067126 047111 041040 052131
11780 067134 027105 000040
11781
11782 067140 042600 052116 051105 PDMSG1: .ASCII <CRLF>'ENTERING CACHE ADDRESS MEMORY POWER UP '
11783 067146 047111 020107 040503
11784 067154 044103 020105 042101
11785 067162 051104 051505 020123
11786 067170 042515 047515 054522
11787 067176 050040 053517 051105
11788 067204 052440 020120
11789 067210 047111 040526 044514 .ASCII 'INVALIDATOR TEST.'<CRLF>
11790 067216 040504 047524 020122
11791 067224 042524 052123 100056
11792 067232 046120 040505 042523 .ASCII 'PLEASE GO THROUGH A POWER DOWN, POWER UP '
11793 067240 043440 020117 044124
11794 067246 047522 043525 020110
11795 067254 020101 047520 042527
11796 067262 020122 047504 047127
11797 067270 020054 047520 042527
11798 067276 020122 050125 040
11799 067303 123 050505 042525 .ASCIZ 'SEQUENCE.'<CRLF>
11800 067310 041516 027105 000200
11801
11802 067316 041600 041501 042510 PDMSG2: .ASCII <CRLF>'CACHE ADDRESS MEMORY POWER UP INVALIDATOR'
11803 067324 040440 042104 042522
11804 067332 051523 046440 046505
11805 067340 051117 020131 047520
11806 067346 042527 020122 050125
11807 067354 044440 053116 046101
11808 067362 042111 052101 051117
11809 067370 052040 051505 020124 .ASCIZ ' TEST DID NOT FAIL.'<CRLF>
11810 067376 044504 020104 047516
11811 067404 020124 040506 046111
11812 067412 100056 000
11813
11814 067415 105 051122 051117 ADRNG: .ASCII 'ERROR ADDRESS REGISTER NEEDED FOR TEST.'<CRLF>'BUT IT HAS BEEN '
11815 067422 040440 042104 042522
11816 067430 051523 051040 043505
11817 067436 051511 042524 020122
11818 067444 042516 042105 042105
11819 067452 043040 051117 052040
11820 067460 051505 026124 041200
11821 067466 052125 044440 020124
11822 067474 040510 020123 042502
11823 067502 047105 040
11824 067505 106 040514 043507 .ASCIZ 'FLAGGED AS BAD!'
11825 067512 042105 040440 020123
11826 067520 040502 020504 000
11827
11828 067525 105 051122 051117 ERRNG: .ASCII 'ERROR REGISTER NEEDED FOR TEST.'<CRLF>'BUT IT HAS BEEN '
    
```

11829	067532	051040	043505	051511	
11830	067540	042524	020122	042516	
11831	067546	042105	042105	043040	
11832	067554	051117	052040	051505	
11833	067562	026124	041200	052125	
11834	067570	044440	020124	040510	
11835	067576	020123	042502	047105	
11836	067604	040			
11837	067605	106	040514	043507	.ASCIZ 'FLAGGED AS BAD!'
11838	067612	042105	040440	020123	
11839	067620	040502	020504	000	
11840					
11841	067625	103	047117	051124	CNRNG: .ASCII 'CONTROL REGISTER NEEDED FOR TEST.' <crlf>'BUT IT HAS BEEN '</crlf>
11842	067632	046117	051040	043505	
11843	067640	051511	042524	020122	
11844	067646	042516	042105	042105	
11845	067654	043040	051117	052040	
11846	067662	051505	026124	041200	
11847	067670	052125	044440	020124	
11848	067676	040510	020123	042502	
11849	067704	047105	040		
11850	067707	106	040514	043507	.ASCIZ 'FLAGGED AS BAD!'
11851	067714	042105	040440	020123	
11852	067722	040502	020504	000	
11853	067727	115	044501	052116	MNRNG: .ASCII 'MAINTENANCE REGISTER NEEDED FOR TEST.' <crlf>'BUT IT HAS BEEN '</crlf>
11854	067734	047105	047101	042503	
11855	067742	051040	043505	051511	
11856	067750	042524	020122	042516	
11857	067756	042105	042105	043040	
11858	067764	051117	052040	051505	
11859	067772	026124	041200	052125	
11860	070000	044440	020124	040510	
11861	070006	020123	042502	047105	
11862	070014	040			
11863	070015	106	040514	043507	.ASCIZ 'FLAGGED AS BAD!'
11864	070022	042105	040440	020123	
11865	070030	040502	020504	000	
11866					
11867	070035	110	052111	046457	HMRNG: .ASCII 'HIT/MISS REGISTER NEEDED FOR TEST.' <crlf>'BUT IT HAS BEEN '</crlf>
11868	070042	051511	020123	042522	
11869	070050	044507	052123	051105	
11870	070056	047040	042505	042504	
11871	070064	020104	047506	020122	
11872	070072	042524	052123	100054	
11873	070100	052502	020124	052111	
11874	070106	044040	051501	041040	
11875	070114	042505	020116		
11876	070120	046106	043501	042507	.ASCIZ 'FLAGGED AS BAD!'
11877	070126	020104	051501	041040	
11878	070134	042101	000041		
11879					
11880	070140	040600	042104	042522	MTA77: .ASCIZ <CRLF>'ADDRESS: '
11881	070146	051523	020072	000040	
11882					
11883	070154	051440	047510	046125	MTB77: .ASCIZ ' SHOULD HAVE BEEN A HIT IN GROUP '
11884	070162	020104	040510	042526	

11885	070170	041040	042505	020116	
11886	070176	020101	044510	020124	
11887	070204	047111	043440	047522	
11888	070212	050125	000040		
11889					
11890	070216	043101	042524	020122	MTC77: .ASCIZ 'AFTER REFERENCING'<CRLF>'ADDRESS: '
11891	070224	042522	042506	042522	
11892	070232	041516	047111	100107	
11893	070240	042101	051104	051505	
11894	070246	035123	020040	000	
11895					
11896	070253	040	044127	046111	MTD77: .ASCIZ ' WHILE FORCING SELECTION OF GROUP '
11897	070260	020105	047506	041522	
11898	070266	047111	020107	042523	
11899	070274	042514	052103	047511	
11900	070302	020116	043117	043440	
11901	070310	047522	050125	000040	
11902					
11903	070316	040600	051122	051117	MTA101: .ASCII <CRLF>'ARROR ADRS REG.'<TAB>'ERROR REG.'<TAB>
11904	070324	040440	051104	020123	
11905	070332	042522	027107	042411	
11906	070340	051122	051117	051040	
11907	070346	043505	004456		
11908	070352	054105	042520	052103	.ASCIZ 'EXPECTED ERR.'<TAB>'PATTERN PUT IN MAINT REG.'<CRLF>
11909	070360	042105	042440	051122	
11910	070366	004456	040520	052124	
11911	070374	051105	020116	052520	
11912	070402	020124	047111	046440	
11913	070410	044501	052116	051040	
11914	070416	043505	100056	000	
11915					
11916	070423	200	043101	042524	MTA120: .ASCIZ <CRLF>'AFTER 2ND CYCLE READ '
11917	070430	020122	047062	020104	
11918	070436	054503	046103	020105	
11919	070444	042522	042101	020040	
11920	070452	000			
11921					
11922	070453	200	043101	042524	MTB120: .ASCIZ <CRLF>'AFTER 4TH CYCLE READ '
11923	070460	020122	052064	020110	
11924	070466	054503	046103	020105	
11925	070474	042522	042101	020040	
11926	070502	000			
11927					
11928	070503	200	043101	042524	MTC120: .ASCIZ <CRLF>'AFTER 6TH CYCLE READ '
11929	070510	020122	052066	020110	
11930	070516	054503	046103	020105	
11931	070524	042522	042101	020040	
11932	070532	000			
11933	070533	200	043101	042524	MTD120: .ASCIZ <CRLF>'AFTER 8TH CYCLE READ '
11934	070540	020122	052070	020110	
11935	070546	054503	046103	020105	
11936	070554	042522	042101	020040	
11937	070562	000			
11938					
11939	070563	200	043101	042524	MTE120: .ASCIZ <CRLF>'AFTER 10TH CYCLE READ '
11940	070570	020122	030061	044124	

11941	070576	041440	041531	042514	
11942	070604	051040	040505	020104	
11943	070612	000			
11944					
11945	070613	200	043101	042524	MTF120: .ASCIZ <CRLF>'AFTER 12TH CYCLE READ '
11946	070620	020122	031061	044124	
11947	070626	041440	041531	042514	
11948	070634	051040	040505	020104	
11949	070642	000			
11950					
11951	070643	106	047522	020115	MTG120: .ASCIZ 'FROM THE HIT/MISS REG. EXPECTED '
11952	070650	044124	020105	044510	
11953	070656	027524	044515	051523	
11954	070664	051040	043505	020056	
11955	070672	054105	042520	052103	
11956	070700	042105	000040		
11957					
11958	070704	052200	042510	050040	MTA124: .ASCII <CRLF>'THE PATTERN BEING USED IN THE MAINTENANCE '
11959	070712	052101	042524	047122	
11960	070720	041040	044505	043516	
11961	070726	052440	042523	020104	
11962	070734	047111	052040	042510	
11963	070742	046440	044501	052116	
11964	070750	047105	047101	042503	
11965	070756	040			
11966	070757	122	043505	051511	.ASCIZ 'REGISTER WAS: '
11967	070764	042524	020122	040527	
11968	070772	035123	000040		
11969					
11970	070776	051200	043105	051105	MTA126: .ASCIZ <CRLF>'REFERENCED ADDRESS:'<TAB>
11971	071004	047105	042503	020104	
11972	071012	042101	051104	051505	
11973	071020	035123	000011		
11974					
11975	071024	040600	051122	051117	MTB126: .ASCIZ <CRLF>'ERROR ADDRESS REGISTER:'<TAB>
11976	071032	040440	042104	042522	
11977	071040	051523	051040	043505	
11978	071046	051511	042524	035122	
11979	071054	000011			
11980					
11981	071056	050200	052101	042524	MTA131: .ASCIZ <CRLF>'PATTERN BEING USED IN THE MAINTENANCE REGISTER:'<TAB>
11982	071064	047122	041040	044505	
11983	071072	043516	052440	042523	
11984	071100	020104	047111	052040	
11985	071106	042510	046440	044501	
11986	071114	052116	047105	047101	
11987	071122	042503	051040	043505	
11988	071130	051511	042524	035122	
11989	071136	000011			
11990					
11991	071140	042600	050130	041505	MTB131: .ASCIZ <CRLF>'EXPECTED ERROR REGISTER:'<TAB>
11992	071146	042524	020104	051105	
11993	071154	047522	020122	042522	
11994	071162	044507	052123	051105	
11995	071170	004472	000		
11996					

```

11997 071173 200 047507 020124 MTC131: .ASCIZ <CRLF>'GOT ERROR REGISTER:'<TAB>
11998 071200 051105 047522 020122
11999 071206 042522 044507 052123
12000 071214 051105 004472 000
12001
12002 071221 200 051105 047522 MTA134: .ASCIZ <CRLF>'ERROR ADR REG.'<TAB>'ERROR REG.'<CRLF>
12003 071226 020122 042101 020122
12004 071234 042522 027107 042411
12005 071242 051122 051117 051040
12006 071250 043505 100056 000
12007
12008 071255 200 054105 042520 MTA135: .ASCIZ <CRLF>'EXPECTED ERROR REG.: '
12009 071262 052103 042105 042440
12010 071270 051122 051117 051040
12011 071276 043505 035056 020040
12012 071304 000
12013
12014 071305 107 052117 042440 MTB135: .ASCIZ 'GOT ERROR REG.: '
12015 071312 051122 051117 051040
12016 071320 043505 035056 020040
12017 071326 000
12018
12019 071327 200 054105 042520 MTC135: .ASCIZ <CRLF>'EXPECTED ERROR ADR REG.: '
12020 071334 052103 042105 042440
12021 071342 051122 051117 040440
12022 071350 051104 051040 043505
12023 071356 035056 020040 000
12024
12025 071363 107 052117 042440 MTD135: .ASCIZ 'GOT ERROR ADR REG.: '
12026 071370 051122 051117 040440
12027 071376 051104 051040 043505
12028 071404 035056 020040 000
12029
12030
12031 071411 015 053412 051101 MS01: .ASCII <15><12>/WARNING- THE SIZE OF MEMORY IS DIFFERENT THEN THAT/<CRLF>
12032 071416 044516 043516 020055
12033 071424 044124 020105 044523
12034 071432 042532 047440 020106
12035 071440 042515 047515 054522
12036 071446 044440 020123 044504
12037 071454 043106 051105 047105
12038 071462 020124 044124 047105
12039 071470 052040 040510 100124
12040 071476 044440 042116 041511 .ASCIZ / INDICATED BY THE SYSTEM SIZE REGISTER./
12041 071504 052101 042105 041040
12042 071512 020131 044124 020105
12043 071520 054523 052123 046505
12044 071526 051440 055111 020105
12045 071534 042522 044507 052123
12046 071542 051105 000056
12047 071546 005015 044523 042532 MS02: .ASCIZ <15><12>/SIZE REG. ACTUAL/
12048 071554 051040 043505 020056
12049 071562 020040 020040 041501
12050 071570 052524 046101 000
12051 071575 040 020040 020040 MS03: .ASCIZ / /
12052 071602 020040 000
    
```

12053	071605	200	050103	020125	MSG1:	.ASCIZ<CRLF>	'CPU UNDER TEST FOUND TO BE A ''
12054	071612	047125	042504	020122			
12055	071620	042524	052123	043040			
12056	071626	052517	042116	052040			
12057	071634	020117	042502	040440			
12058	071642	000040					
12059	071644	041113	030461	042455	MSG2:	.ASCIZ	'KB11-EM'<CRLF>
12060	071652	100115	000				
12061	071655	113	030502	026461	MSG3:	.ASCIZ	'KB11-B/C'<CRLF>
12062	071662	027502	100103	000			
12063	071667	113	030502	026461	MSG4:	.ASCIZ	'KB11-CM
12064	071674	046503	020040	020040			'<CRLF>
12065	071702	020040	020040	020040			
12066	071710	020040	020040	020040			
12067	071716	000200					
12068	071720	041113	030461	042455	MSG5:	.ASCIZ	'KB11-E'<CRLF>
12069	071726	000200					
12070	071730	005015	047516	046440	EM724:	.ASCIZ	<CR><LF>/NO MAP REGISTERS AVAILABLE FOR UNIBUS PARITY ERROR TEST/
12071	071736	050101	051040	043505			
12072	071744	051511	042524	051522			
12073	071752	040440	040526	046111			
12074	071760	041101	042514	043040			
12075	071766	051117	052440	044516			
12076	071774	052502	020123	040520			
12077	072002	044522	054524	042440			
12078	072010	051122	051117	052040			
12079	072016	051505	000124				
12080							
12081							
12082							
12083	072022	020101	042522	042506	EM1:	.ASCIZ	'A REFERENCE WHICH SHOULD HAVE BEEN A HIT WAS A MISS.'
12084	072030	042522	041516	020105			
12085	072036	044127	041511	020110			
12086	072044	044123	052517	042114			
12087	072052	044040	053101	020105			
12088	072060	042502	047105	040440			
12089	072066	044040	052111	053440			
12090	072074	051501	040440	046440			
12091	072102	051511	027123	000			
12092							
12093	072107	125	042516	050130	EM2:	.ASCII	'UNEXPECTED ERROR DURING WORST CASE NOISE TEST ON '
12094	072114	041505	042524	020104			
12095	072122	051105	047522	020122			
12096	072130	052504	044522	043516			
12097	072136	053440	051117	052123			
12098	072144	041440	051501	020105			
12099	072152	047516	051511	020105			
12100	072160	042524	052123	047440			
12101	072166	020116					
12102	072170	040503	044103	020105			.ASCII 'CACHE DATA MEMORY.'<CRLF>
12103	072176	040504	040524	046440			
12104	072204	046505	051117	027131			
12105	072212	200					
12106	072213	101	047040	047117			.ASCIZ 'A NON-CACHE DATA PARITY ERROR OCCURRED WHILE TESTING.'
12107	072220	041455	041501	042510			
12108	072226	042040	052101	020101			

;THESE ARE THE ERROR MESSAGES:

12109	072234	040520	044522	054524	
12110	072242	042440	051122	051117	
12111	072250	047440	041503	051125	
12112	072256	042522	020104	044127	
12113	072264	046111	020105	042524	
12114	072272	052123	047111	027107	
12115	072300	000			
12116					
12117	072301	127	051117	052123	EM3: .ASCII 'WORST CASE NOISE TEST OF THE CACHE DATA MEMORY '
12118	072306	041440	051501	020105	
12119	072314	047516	051511	020105	
12120	072322	042524	052123	047440	
12121	072330	020106	044124	020105	
12122	072336	040503	044103	020105	
12123	072344	040504	040524	046440	
12124	072352	046505	051117	020131	
12125	072360	043200	044501	042514	.ASCIZ <CRLF>/FAILED WHILE GALLOPING 0'S./
12126	072366	020104	044127	046111	
12127	072374	020105	040507	046114	
12128	072402	050117	047111	020107	
12129	072410	023460	027123	000	
12130					
12131	072415	127	051117	052123	EM4: .ASCII 'WORST CASE NOISE TEST OF THE CACHE DATA MEMORY'
12132	072422	041440	051501	020105	
12133	072430	047516	051511	020105	
12134	072436	042524	052123	047440	
12135	072444	020106	044124	020105	
12136	072452	040503	044103	020105	
12137	072460	040504	040524	046440	
12138	072466	046505	051117	131	
12139	072473	200	040506	046111	.ASCIZ <CRLF>/FAILED WHILE GALLOPING 1'S./
12140	072500	042105	053440	044510	
12141	072506	042514	043440	046101	
12142	072514	047514	044520	043516	
12143	072522	030440	051447	000056	
12144					
12145	072530	042103	054115	052040	EM5: .ASCIZ 'CDMX TEST FAILURE.'<CRLF>'BAD CACHE GROUP 0 DATA READ.'
12146	072536	051505	020124	040506	
12147	072544	046111	051125	027105	
12148	072552	041200	042101	041440	
12149	072560	041501	042510	043440	
12150	072566	047522	050125	030040	
12151	072574	042040	052101	020101	
12152	072602	042522	042101	000056	
12153					
12154	072610	042103	054115	052040	EM6: .ASCIZ 'CDMX TEST FAILURE.'<CRLF>'BAD CACHE GROUP 1 DATA READ.'
12155	072616	051505	020124	040506	
12156	072624	046111	051125	027105	
12157	072632	041200	042101	041440	
12158	072640	041501	042510	043440	
12159	072646	047522	050125	030440	
12160	072654	042040	052101	020101	
12161	072662	042522	042101	000056	
12162					
12163	072670	042103	054115	052040	EM7: .ASCII 'CDMX TEST FAILURE.'<CRLF>'BAD MAIN MEMORY, EVEN WORD.'
12164	072676	051505	020124	040506	

12165	072704	046111	051125	027105	
12166	072712	041200	042101	046440	
12167	072720	044501	020116	042515	
12168	072726	047515	054522	020054	
12169	072734	053105	047105	053440	
12170	072742	051117	026104		
12171	072746	042040	052101	020101	.ASCIZ ' DATA READ.'
12172	072754	042522	042101	000056	
12173					
12174	072762	042103	054115	052040	EM10: .ASCII 'CDMX TEST FAILURE.' <crlf&gt;'bad main="" memory,="" odd="" td="" word.'<=""></crlf&gt;'bad>
12175	072770	051505	020124	040506	
12176	072776	046111	051125	027105	
12177	073004	041200	042101	046440	
12178	073012	044501	020116	042515	
12179	073020	047515	054522	020054	
12180	073026	042117	020104	047527	
12181	073034	042122	054		
12182	073037	040	040504	040524	.ASCIZ ' DATA READ.'
12183	073044	051040	040505	027104	
12184	073052	000			
12185					
12186	073053	120	051101	052111	EM11: .ASCIZ 'PARITY ERROR IN CACHE DATA MEMORY COUNT PATTERN TEST.'
12187	073060	020131	051105	047522	
12188	073066	020122	047111	041440	
12189	073074	041501	042510	042040	
12190	073102	052101	020101	042515	
12191	073110	047515	054522	041440	
12192	073116	052517	052116	050040	
12193	073124	052101	042524	047122	
12194	073132	052040	051505	027124	
12195	073140	000			
12196					
12197	073141	102	042101	042040	EM12: .ASCII 'BAD DATA WAS READ IN CACHE MEMORY COUNT PATTERN '
12198	073146	052101	020101	040527	
12199	073154	020123	042522	042101	
12200	073162	044440	020116	040503	
12201	073170	044103	020105	042515	
12202	073176	047515	054522	041440	
12203	073204	052517	052116	050040	
12204	073212	052101	042524	047122	
12205	073220	040			
12206	073221	124	051505	027124	.ASCIZ 'TEST.' <crlf&gt;'but abort="" no="" occurred.'<="" or="" td="" trap=""></crlf&gt;'but>
12207	073226	041200	052125	047040	
12208	073234	020117	051124	050101	
12209	073242	047440	020122	041101	
12210	073250	051117	020124	041517	
12211	073256	052503	051122	042105	
12212	073264	000056			
12213					
12214	073266	040503	044103	020105	EM13: .ASCII 'CACHE MEMORY COUNT PATTERN TEST.' <crlf&gt;< td=""></crlf&gt;<>
12215	073274	042515	047515	054522	
12216	073302	041440	052517	052116	
12217	073310	050040	052101	042524	
12218	073316	047122	052040	051505	
12219	073324	027124	200		
12220	073327	105	051122	051117	.ASCIZ 'ERROR SUMMARY.'



12221	073334	051440	046525	040515	
12222	073342	054522	000056		
12223					
12224	073346	052600	042516	050130	EM14: .ASCIZ <CRLF>'UNEXPECTED PARITY ERROR TRAP.'
12225	073354	041505	042524	020104	
12226	073362	040520	044522	054524	
12227	073370	042440	051122	051117	
12228	073376	052040	040522	027120	
12229	073404	000			
12230					
12231	073405	052	025052	042524	EM15: .ASCIZ '***TEST ABORTED! GOING TO NEXT TEST.***'
12232	073412	052123	040440	047502	
12233	073420	052122	042105	020041	
12234	073426	047507	047111	020107	
12235	073434	047524	047040	054105	
12236	073442	020124	042524	052123	
12237	073450	025056	025052	000	
12238					
12239					
12240	073455	103	041501	042510	EM16: .ASCIZ 'CACHE DATA MEMORY DUAL ADDRESS TEST FAILED.'
12241	073462	042040	052101	020101	
12242	073470	042515	047515	054522	
12243	073476	042040	040525	020114	
12244	073504	042101	051104	051505	
12245	073512	020123	042524	052123	
12246	073520	043040	044501	042514	
12247	073526	027104	000		
12248					
12249	073531	103	041501	042510	EM17: .ASCIZ 'CACHE DATA MEMORY BYTE ENABLE LOGIC TEST FAILED.'
12250	073536	042040	052101	020101	
12251	073544	042515	047515	054522	
12252	073552	041040	052131	020105	
12253	073560	047105	041101	042514	
12254	073566	046040	043517	041511	
12255	073574	052040	051505	020124	
12256	073602	040506	046111	042105	
12257	073610	000056			
12258					
12259		073531			EM20=EM17
12260					
12261	073612	040503	044103	020105	EM21: .ASCIZ 'CACHE DATA MEMORY CHIP SELECTION LOGIC TEST FAILED.'
12262	073620	040504	040524	046440	
12263	073626	046505	051117	020131	
12264	073634	044103	050111	051440	
12265	073642	046105	041505	044524	
12266	073650	047117	046040	043517	
12267	073656	041511	052040	051505	
12268	073664	020124	040506	046111	
12269	073672	042105	000056		
12270					
12271	073676	042101	051104	051505	EM22: .ASCII 'ADDRESS MULTIPLEXER TEST WAS UNABLE TO FORCE'
12272	073704	020123	052515	052114	
12273	073712	050111	042514	042530	
12274	073720	020122	042524	052123	
12275	073726	053440	051501	052440	
12276	073734	040516	046102	020105	

12277 073742 047524 043040 051117  
12278 073750 042503  
12279 073752 040440 050040 051101  
12280 073760 052111 020131 051105  
12281 073766 047522 026122 052440  
12282 073774 044523 043516 052040  
12283 074002 042510 100040  
12284 074006 040515 047111 042524  
12285 074014 040516 041516 020105  
12286 074022 042522 044507 052123  
12287 074030 051105 020054 047117  
12288 074036 052040 042510  
12289 074042 046440 044501 020116  
12290 074050 042515 047515 054522  
12291 074056 040440 042104 042522  
12292 074064 051523 040440 042116  
12293 074072 041440 047117 051124  
12294 074100 046117 046040 047111  
12295 074106 051505 000056  
12296  
12297 074112 042101 051104 051505  
12298 074120 020123 052515 052114  
12299 074126 050111 042514 042530  
12300 074134 026122 040440 054115  
12301 074142 020054 050103 020125  
12302 074150 047111 052520 051524  
12303 074156 052040 051505 020124  
12304 074164 040506 046111 042105  
12305 074172 056  
12306 074173 200 051105 047522  
12307 074200 020122 042101 051104  
12308 074206 051505 020123 042522  
12309 074214 044507 052123 051105  
12310 074222 047040 052117 051440  
12311 074230 052105 041440 051117  
12312 074236 042522 052103 054514  
12313 074244 000056  
12314  
12315 073676  
12316  
12317 074112  
12318  
12319 074246 042101 051104 051505  
12320 074254 020123 042515 047515  
12321 074262 054522 020054 042101  
12322 074270 051104 051505 020123  
12323 074276 047503 050115 051101  
12324 074304 052101 051117 052040  
12325 074312 051505 020124 040506  
12326 074320 046111 051125 027105  
12327 074326 040600 020116 042101  
12328 074334 051104 051505 020123  
12329 074342 044127 041511 020110  
12330 074350 044123 052517 042114  
12331 074356 044040 053101 020105  
12332 074364 042502 047105 040440

.ASCII ' A PARITY ERROR, USING THE '<CRLF>

.ASCII 'MAINTENANCE REGISTER, ON THE'

.ASCIZ ' MAIN MEMORY ADDRESS AND CONTROL LINES.'

EM23: .ASCII 'ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FAILED.'

.ASCIZ <CRLF>'ERROR ADDRESS REGISTER NOT SET CORRECTLY.'

EM24=EM22

EM25=EM23

EM26: .ASCII 'ADDRESS MEMORY, ADDRESS COMPARATOR TEST FAILURE.'

.ASCII <CRLF>'AN ADDRESS WHICH SHOULD HAVE BEEN A HIT WAS'

12333	074372	044040	052111	053440	
12334	074400	051501			
12335	074402	040440	046440	051511	.ASCIZ ' A MISS.'
12336	074410	027123	000		
12337					
12338	074413	101	042104	042522	EM27: .ASCII 'ADDRESS MEMORY, ADDRESS COMPARATOR TEST FAILURE.'
12339	074420	051523	046440	046505	
12340	074426	051117	026131	040440	
12341	074434	042104	042522	051523	
12342	074442	041440	046517	040520	
12343	074450	040522	047524	020122	
12344	074456	042524	052123	043040	
12345	074464	044501	052514	042522	
12346	074472	056			
12347	074473	200	047101	040440	.ASCII <CRLF>'AN ADDRESS WHICH SHOULD HAVE BEEN A MISS '
12348	074500	042104	042522	051523	
12349	074506	053440	044510	044103	
12350	074514	051440	047510	046125	
12351	074522	020104	040510	042526	
12352	074530	041040	042505	020116	
12353	074536	020101	044515	051523	
12354	074544	040			
12355	074545	127	051501	040440	.ASCIZ 'WAS A HIT.'
12356	074552	044040	052111	000056	
12357					
12358		073676			EM30=EM22
12359					
12360	074560	042101	051104	051505	EM31: .ASCII 'ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FAILED.'
12361	074566	020123	052515	052114	
12362	074574	050111	042514	042530	
12363	074602	026122	040440	054115	
12364	074610	020054	047125	041111	
12365	074616	051525	044440	050116	
12366	074624	052125	020123	042524	
12367	074632	052123	043040	044501	
12368	074640	042514	027104		
12369	074644	042600	051122	051117	.ASCIZ <CRLF>'ERROR ADDRESS REGISTER NOT SET CORRECTLY.'
12370	074652	040440	042104	042522	
12371	074660	051523	051040	043505	
12372	074666	051511	042524	020122	
12373	074674	047516	020124	042523	
12374	074702	020124	047503	051122	
12375	074710	041505	046124	027131	
12376	074716	000			
12377					
12378		073676			EM32=EM22
12379					
12380		074560			EM33=EM31
12381					
12382	074717	101	042104	042522	EM34: .ASCII 'ADDRESS MULTIPLEXER, AMX, DUAL ADDRESS TEST,'<CRLF>
12383	074724	051523	046440	046125	
12384	074732	044524	046120	054105	
12385	074740	051105	020054	046501	
12386	074746	026130	042040	040525	
12387	074754	020114	042101	051104	
12388	074762	051505	020123	042524	

12389	074770	052123	100054		
12390	074774	047117	041440	052520	.ASCIZ 'ON CPU INPUTS, FAILED.'
12391	075002	044440	050116	052125	
12392	075010	026123	043040	044501	
12393	075016	042514	027104	000	
12394					
12395	075023	101	042104	042522	EM35: .ASCII 'ADDRESS MULTIPLEXER, AMX, DUAL ADDRESS TEST,'<CRLF>
12396	075030	051523	046440	046125	
12397	075036	044524	046120	054105	
12398	075044	051105	020054	046501	
12399	075052	026130	042040	040525	
12400	075060	020114	042101	051104	
12401	075066	051505	020123	042524	
12402	075074	052123	100054		
12403	075100	047117	052440	044516	.ASCIZ 'ON UNIBUS INPUTS, FAILED.'
12404	075106	052502	020123	047111	
12405	075114	052520	051524	020054	
12406	075122	040506	046111	042105	
12407	075130	000056			
12408					
12409	075132	042101	051104	051505	EM36: .ASCII 'ADDRESS MEMORY COUNT PATTERN TEST FAILURE,'<CRLF>
12410	075140	020123	042515	047515	
12411	075146	054522	041440	052517	
12412	075154	052116	050040	052101	
12413	075162	042524	047122	052040	
12414	075170	051505	020124	040506	
12415	075176	046111	051125	026105	
12416	075204	200			
12417	075205	116	020117	040520	.ASCIZ 'NO PARITY ERROR OCCURS, BUT CAN NOT GET A HIT.'
12418	075212	044522	054524	042440	
12419	075220	051122	051117	047440	
12420	075226	041503	051125	026123	
12421	075234	041040	052125	041440	
12422	075242	047101	047040	052117	
12423	075250	043440	052105	040440	
12424	075256	044040	052111	000056	
12425					
12426	075264	042101	051104	051505	EM37: .ASCIZ 'ADDRESS MEMORY COUNT PATTERN TEST, ERROR SUMMARY.'
12427	075272	020123	042515	047515	
12428	075300	054522	041440	052517	
12429	075306	052116	050040	052101	
12430	075314	042524	047122	052040	
12431	075322	051505	026124	042440	
12432	075330	051122	051117	051440	
12433	075336	046525	040515	054522	
12434	075344	000056			
12435					
12436	075346	042101	051104	051505	EM40: .ASCII 'ADDRESS MEMORY COUNT PATTERN TEST FAILURE,'<CRLF>
12437	075354	020123	042515	047515	
12438	075362	054522	041440	052517	
12439	075370	052116	050040	052101	
12440	075376	042524	047122	052040	
12441	075404	051505	020124	040506	
12442	075412	046111	051125	026105	
12443	075420	200			
12444	075421	103	041501	042510	.ASCII 'CACHE MEMORY ADDRESS PARITY ERROR OCCURRED'

12445	075426	046440	046505	051117	
12446	075434	020131	042101	051104	
12447	075442	051505	020123	040520	
12448	075450	044522	054524	042440	
12449	075456	051122	051117	047440	
12450	075464	041503	051125	042522	
12451	075472	104			
12452	075473	040	052101	052040	.ASCIZ ' AT THE TEST ADDRESS.'
12453	075500	042510	052040	051505	
12454	075506	020124	042101	051104	
12455	075514	051505	027123	000	
12456					
12457	075521	101	042104	042522	EM41: .ASCII 'ADDRESS MEMORY DUAL ADDRESS TEST FAILED TO GET '
12458	075526	051523	046440	046505	
12459	075534	051117	020131	052504	
12460	075542	046101	040440	042104	
12461	075550	042522	051523	052040	
12462	075556	051505	020124	040506	
12463	075564	046111	042105	052040	
12464	075572	020117	042507	020124	
12465	075600	020101	044510	020124	.ASCII 'A HIT AT A TEST ADDRESS,'<CRLF>
12466	075606	052101	040440	052040	
12467	075614	051505	020124	042101	
12468	075622	051104	051505	026123	
12469	075630	200			
12470	075631	127	044510	042514	.ASCIZ 'WHILE WRITING THE ADDRESS MEMORY LOCATIONS.'
12471	075636	053440	044522	044524	
12472	075644	043516	052040	042510	
12473	075652	040440	042104	042522	
12474	075660	051523	046440	046505	
12475	075666	051117	020131	047514	
12476	075674	040503	044524	047117	
12477	075702	027123	000		
12478					
12479	075705	101	042104	042522	EM42: .ASCII 'ADDRESS MEMORY DUAL ADDRESS TEST FAILED TO GET'
12480	075712	051523	046440	046505	
12481	075720	051117	020131	052504	
12482	075726	046101	040440	042104	
12483	075734	042522	051523	052040	
12484	075742	051505	020124	040506	
12485	075750	046111	042105	052040	
12486	075756	020117	042507	124	
12487	075763	101	044040	052111	.ASCII 'A HIT AT A TEST ADDRESS,'<CRLF>
12488	075770	040440	020124	020101	
12489	075776	042524	052123	040440	
12490	076004	042104	042522	051523	
12491	076012	100054			
12492					
12493	076014	044127	046111	020105	.ASCII 'WHILE READING BACK THE ADDRESS MEMORY LOCATIONS.'<CRLF><LF>
12494	076022	042522	042101	047111	
12495	076030	020107	040502	045503	
12496	076036	052040	042510	040440	
12497	076044	042104	042522	051523	
12498	076052	046440	046505	051117	
12499	076060	020131	047514	040503	
12500	076066	044524	047117	027123	

12501	076074	005200				.ASCIZ '[THIS PROBLEM MIGHT BE CORRECTED BY ECO M8182-4]'	<CRLF>
12502	076076	052133	044510	020123			
12503	076104	051120	041117	042514			
12504	076112	020115	044515	044107			
12505	076120	020124	042502	041440			
12506	076126	051117	042522	052103			
12507	076134	042105	041040	020131			
12508	076142	041505	020117	034115			
12509	076150	034061	026462	056464			
12510	076156	000200					
12511							
12512							
12513							
12514	076160	042101	051104	051505	EM43:	.ASCII 'ADDRESS MEMORY DUAL ADDRESS TEST FAILURE,'	<CRLF>
12515	076166	020123	042515	047515			
12516	076174	054522	042040	040525			
12517	076202	020114	042101	051104			
12518	076210	051505	020123	042524			
12519	076216	052123	043040	044501			
12520	076224	052514	042522	100054		.ASCIZ 'CACHE ADDRESS MEMORY PARITY ERROR OCCURRED.'	
12521	076232	040503	044103	020105			
12522	076240	042101	051104	051505			
12523	076246	020123	042515	047515			
12524	076254	054522	050040	051101			
12525	076262	052111	020131	051105			
12526	076270	047522	020122	041517			
12527	076276	052503	051122	042105			
12528	076304	000056					
12529							
12530							
12531	076306	040515	047111	046440	EM44:	.ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED,'	
12532	076314	046505	051117	020131			
12533	076322	054502	042524	046440			
12534	076330	051501	020113	042507			
12535	076336	042516	0405??	047524			
12536	076344	020122	042524	052123			
12537	076352	043040	044501	042514			
12538	076360	026104					
12539	076362	042040	044517	043516		.ASCII ' DOING CPU DATOB.'	<CRLF>
12540	076370	041440	052520	042040			
12541	076376	052101	041117	100056		.ASCII 'A MAIN MEMORY ADDRESS AND CONTROL LINE '	
12542	076404	020101	040515	047111			
12543	076412	046440	046505	051117			
12544	076420	020131	042101	051104			
12545	076426	051505	020123	047101			
12546	076434	020104	047503	052116			
12547	076442	047522	020114	044514			
12548	076450	042516	040				
12549	076453	120	051101	052111		.ASCIZ 'PARITY ERROR OCCURRED.'	
12550	076460	020131	051105	047522			
12551	076466	020122	041517	052503			
12552	076474	051122	042105	000056			
12553							
12554	076502	040515	047111	046440	EM45:	.ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED,'	
12555	076510	046505	051117	020131			
12556	076516	054502	042524	046440			

12557	076524	051501	020113	042507	
12558	076532	042516	040522	047524	
12559	076540	020122	042524	052123	
12560	076546	043040	044501	042514	
12561	076554	026104			
12562	076556	042040	044517	043516	.ASCII ' DOING UNIBUS DATOB.'<CRLF>
12563	076564	052440	044516	052502	
12564	076572	020123	040504	047524	
12565	076600	027102	200		
12566	076603	101	046440	044501	.ASCII 'A MAIN MEMORY ADDRESS AND CONTROL LINE '
12567	076610	020116	042515	047515	
12568	076616	054522	040440	042104	
12569	076624	042522	051523	040440	
12570	076632	042116	041440	047117	
12571	076640	051124	046117	046040	
12572	076646	047111	020105		
12573	076652	040520	044522	054524	.ASCIZ 'PARITY ERROR OCCURRED.'
12574	076660	042440	051122	051117	
12575	076666	047440	041503	051125	
12576	076674	042522	027104	000	
12577					
12578	076701	115	044501	020116	EM46: .ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED.'
12579	076706	042515	047515	054522	
12580	076714	041040	052131	020105	
12581	076722	040515	045523	043440	
12582	076730	047105	051105	052101	
12583	076736	051117	052040	051505	
12584	076744	020124	040506	046111	
12585	076752	042105	056		
12586	076755	200	051127	047117	.ASCIZ <CRLF>'WRONG BYTE WRITTEN, ON A CPU DATOB.'
12587	076762	020107	054502	042524	
12588	076770	053440	044522	052124	
12589	076776	047105	020054	047117	
12590	077004	040440	041440	052520	
12591	077012	042040	052101	041117	
12592	077020	000056			
12593					
12594	077022	040515	047111	046440	EM47: .ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED.'
12595	077030	046505	051117	020131	
12596	077036	054502	042524	046440	
12597	077044	051501	020113	042507	
12598	077052	042516	040522	047524	
12599	077060	020122	042524	052123	
12600	077066	043040	044501	042514	
12601	077074	027104			
12602	077076	053600	047522	043516	.ASCIZ <CRLF>'WRONG BYTE WRITTEN, ON A UNIBUS DATOB.'
12603	077104	041040	052131	020105	
12604	077112	051127	052111	042524	
12605	077120	026116	047440	020116	
12606	077126	020101	047125	041111	
12607	077134	051525	042040	052101	
12608	077142	041117	000056		
12609					
12610	076306				EM50=EM44
12611					
12612	076502				EM51=EM45

12613						
12614		076701				EM52=EM46
12615						
12616		077022				EM53=EM47
12617						
12618	077146	040503	044103	020105		EM54: .ASCII 'CACHE ADDRESS MEMORY POWER UP INVALIDATOR TEST FAILED.'
12619	077154	042101	051104	051505		
12620	077162	020123	042515	047515		
12621	077170	054522	050040	053517		
12622	077176	051105	052440	020120		
12623	077204	047111	040526	044514		
12624	077212	040504	047524	020122		
12625	077220	042524	052123	043040		
12626	077226	044501	042514	027104		
12627	077234	041600	041501	042510		.ASCII <CRLF>'CACHE DATA OR ADDRESS MEMORY PARITY '
12628	077242	042040	052101	020101		
12629	077250	051117	040440	042104		
12630	077256	042522	051523	046440		
12631	077264	046505	051117	020131		
12632	077272	040520	044522	054524		
12633	077300	040				
12634	077301	105	051122	051117		.ASCIZ 'ERROR DETECTED.'
12635	077306	042040	052105	041505		
12636	077314	042524	027104	000		
12637	077321	103	051103	041440		EM55: .ASCIZ /CCR COULD NOT BE CLEARED/
12638	077326	052517	042114	047040		
12639	077334	052117	041040	020105		
12640	077342	046103	040505	042522		
12641	077350	000104				
12642	077352	053111	051523	024040		EM56: .ASCIZ /IVSS (BIT 14) COULD NOT BE SET IN CCR/
12643	077360	044502	020124	032061		
12644	077366	020051	047503	046125		
12645	077374	020104	047516	020124		
12646	077402	042502	051440	052105		
12647	077410	044440	020116	041503		
12648	077416	000122				
12649	077420	053111	051523	041440		EM57: .ASCIZ /IVSS COULD NOT BE CLEARED IN CCR/
12650	077426	052517	042114	047040		
12651	077434	052117	041040	020105		
12652	077442	046103	040505	042522		
12653	077450	020104	047111	041440		
12654	077456	051103	000			
12655	077461	126	044523	020125		EM60: .ASCIZ /VSIU (BIT 13) COULD NOT BE SET/
12656	077466	041050	052111	030440		
12657	077474	024463	041440	052517		
12658	077502	042114	047040	052117		
12659	077510	041040	020105	042523		
12660	077516	000124				
12661	077520	041526	050111	042040		EM61: .ASCIZ /VCIP DID NOT CLEAR AFTER CACHE FLUSH (ON SETTING VSIU)/
12662	077526	042111	047040	052117		
12663	077534	041440	042514	051101		
12664	077542	040440	052106	051105		
12665	077550	041440	041501	042510		
12666	077556	043040	052514	044123		
12667	077564	024040	047117	051440		
12668	077572	052105	044524	043516		



12669	077600	053040	044523	024525	
12670	077606	000			
12671	077607	126	044523	020125	EM62: .ASCIZ /VSIU COULD NOT BE CLEARED/
12672	077614	047503	046125	020104	
12673	077622	047516	020124	042502	
12674	077630	041440	042514	051101	
12675	077636	042105	000		
12676	077641	126	044503	020120	EM63: .ASCIZ /VCIP DID NOT SET WHEN CACHE FLUSH BIT WAS SET/
12677	077646	044504	020104	047516	
12678	077654	020124	042523	020124	
12679	077662	044127	047105	041440	
12680	077670	041501	042510	043040	
12681	077676	052514	044123	041040	
12682	077704	052111	053440	051501	
12683	077712	051440	052105	000	
12684	077717	126	044523	020125	EM64: .ASCIZ /VSIU DID NOT SWITCH WHEN CACHE FLUSH BIT WAS SET/
12685	077724	044504	020104	047516	
12686	077732	020124	053523	052111	
12687	077740	044103	053440	042510	
12688	077746	020116	040503	044103	
12689	077754	020105	046106	051525	
12690	077762	020110	044502	020124	
12691	077770	040527	020123	042523	
12692	077776	000124			
12693	100000	051526	052511	051440	EM65: .ASCIZ /VSIU SWITCHED WHEN CACHE FLUSH WAS DONE, WITH IVSS SET/
12694	100006	044527	041524	042510	
12695	100014	020104	044127	047105	
12696	100022	041440	041501	042510	
12697	100030	043040	052514	044123	
12698	100036	053440	051501	042040	
12699	100044	047117	026105	044527	
12700	100052	044124	044440	051526	
12701	100060	020123	042523	000124	
12702	100066	042524	052123	042055	EM66: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12703	100074	052101	020101	042522	
12704	100102	042506	042522	041516	
12705	100110	020105	047516	020124	
12706	100116	020101	044515	051523	
12707	100124	005015	040526	044514	.ASCIZ <15><12>/VALID STORE NOT SWITCHED ON CACHE FLUSH/
12708	100132	020104	052123	051117	
12709	100140	020105	047516	020124	
12710	100146	053523	052111	044103	
12711	100154	042105	047440	020116	
12712	100162	040503	044103	020105	
12713	100170	046106	051525	000110	
12714	100176	042524	052123	042055	EM67: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12715	100204	052101	020101	042522	
12716	100212	042506	042522	041516	
12717	100220	020105	047516	020124	
12718	100226	020101	044515	051523	
12719	100234	005015	040526	044514	.ASCIZ <15><12>/VALID STORE NOT INVALIDATED ON CACHE FLUSH/
12720	100242	020104	052123	051117	
12721	100250	020105	047516	020124	
12722	100256	047111	040526	044514	
12723	100264	040504	042524	020104	
12724	100272	047117	041440	041501	

12725	100300	042510	043040	052514	
12726	100306	044123	000		
12727	100311	124	051505	026524	EM70: .ASCII /TEST-DATA REFERENCE NOT A HIT/
12728	100316	040504	040524	051040	
12729	100324	043105	051105	047105	
12730	100332	042503	047040	052117	
12731	100340	040440	044040	052111	
12732	100346	005015	051106	046517	.ASCIZ <15><12>/FROM THE GROUP AND VALID STORE BEING CHECKED/
12733	100354	052040	042510	043440	
12734	100362	047522	050125	040440	
12735	100370	042116	053040	046101	
12736	100376	042111	051440	047524	
12737	100404	042522	041040	044505	
12738	100412	043516	041440	042510	
12739	100420	045503	042105	000	
12740	100425	104	052101	020101	EM71: .ASCIZ /DATA ERROR ON READING CACHED LOCATION/
12741	100432	051105	047522	020122	
12742	100440	047117	051040	040505	
12743	100446	044504	043516	041440	
12744	100454	041501	042510	020104	
12745	100462	047514	040503	044524	
12746	100470	047117	000		
12747	100473	124	051505	026524	EM72: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12748	100500	040504	040524	051040	
12749	100506	043105	051105	047105	
12750	100514	042503	047040	052117	
12751	100522	040440	046440	051511	
12752	100530	123			
12753	100531	015	041412	041501	.ASCIZ <15><12>/CACHE DOES NOT TURN OFF, WHEN FLUSH DONE WITH IVSS SET/
12754	100536	042510	042040	042517	
12755	100544	020123	047516	020124	
12756	100552	052524	047122	047440	
12757	100560	043106	020054	044127	
12758	100566	047105	043040	052514	
12759	100574	044123	042040	047117	
12760	100602	020105	044527	044124	
12761	100610	044440	051526	020123	
12762	100616	042523	000124		
12763	100622	042524	052123	042055	EM73: .ASCII /TEST-DATA REFERENCE NOT A HIT/
12764	100630	052101	020101	042522	
12765	100636	042506	042522	041516	
12766	100644	020105	047516	020124	
12767	100652	020101	044510	124	
12768	100657	015	041412	041501	.ASCIZ <15><12>/CACHE DOES NOT TURN ON AFTER TURNING OFF/
12769	100664	042510	042040	042517	
12770	100672	020123	047516	020124	
12771	100700	052524	047122	047440	
12772	100706	020116	043101	042524	
12773	100714	020122	052524	047122	
12774	100722	047111	020107	043117	
12775	100730	000106			
12776	100732	042524	052123	042055	EM74: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12777	100740	052101	020101	042522	
12778	100746	042506	042522	041516	
12779	100754	020105	047516	020124	
12780	100762	020101	044515	051523	

12781	100770	005015	040503	044103
12782	100776	020105	054502	040520
12783	101004	051523	042040	042111
12784	101012	047040	052117	043040
12785	101020	051117	042503	040440
12786	101026	046440	051511	000123
12787	101034	042524	052123	042055
12788	101042	052101	020101	042522
12789	101050	042506	042522	041516
12790	101056	020105	047516	020124
12791	101064	020101	044515	051523
12792	101072	005015	040503	044103
12793	101100	020105	054502	040520
12794	101106	051523	042040	042111
12795	101114	047040	052117	044440
12796	101122	053116	046101	042111
12797	101130	052101	020105	040503
12798	101136	044103	042105	042040
12799	101144	052101	000101	
12800	101150	042524	052123	042055
12801	101156	052101	020101	042522
12802	101164	042506	042522	041516
12803	101172	020105	047516	020124
12804	101200	020101	044515	051523
12805	101206	005015	051501	041122
12806	101214	042040	042111	047040
12807	101222	052117	043040	051117
12808	101230	042503	040440	046440
12809	101236	051511	020123	047117
12810	101244	052040	042510	047440
12811	101252	042520	040522	042116
12812	101260	000		
12813	101261	124	051505	026524
12814	101266	040504	040524	051040
12815	101274	043105	051105	047105
12816	101302	042503	047040	052117
12817	101310	040440	046440	051511
12818	101316	123		
12819	101317	015	041412	041501
12820	101324	042510	020104	050117
12821	101332	051105	047101	020104
12822	101340	047516	020124	047111
12823	101346	040526	044514	040504
12824	101354	042524	020104	047117
12825	101362	040440	051123	020102
12826	101370	054105	041505	052125
12827	101376	047511	100116	
12828	101402	052133	044510	020123
12829	101410	051120	041117	042514
12830	101416	020115	044515	044107
12831	101424	020124	042502	041440
12832	101432	051117	042522	052103
12833	101440	042105	041040	020131
12834	101446	041505	020117	034115
12835	101454	034061	026462	056464
12836	101462	000200		

.ASCIZ <15><12>/CACHE BYPASS DID NOT FORCE A MISS/

EM75: .ASCII /TEST-DATA REFERENCE NOT A MISS/

.ASCIZ <15><12>/CACHE BYPASS DID NOT INVALIDATE CACHED DATA/

EM76: .ASCII /TEST-DATA REFERENCE NOT A MISS/

.ASCIZ <15><12>/ASRB DID NOT FORCE A MISS ON THE OPERAND/

EM77: .ASCII /TEST-DATA REFERENCE NOT A MISS/

.ASCII <15><12>/CACHED OPERAND NOT INVALIDATED ON ASRB EXECUTION/<CRLF>

.ASCIZ /[THIS PROBLEM MIGHT BE CORRECTED BY ECO M8182-4]/<CRLF>

12837	101464	042524	052123	042055	EM100: .ASCIZ /TEST-DATA COULD NOT BE MADE HIT/
12838	101472	052101	020101	047503	
12839	101500	046125	020104	047516	
12840	101506	020124	042502	046440	
12841	101514	042101	020105	044510	
12842	101522	000124			
12843	101524	047516	050040	051101	EM103: .ASCIZ /NO PARITY ERROR TRAP ON VALID STORE PARITY ERROR/
12844	101532	052111	020131	051105	
12845	101540	047522	020122	051124	
12846	101546	050101	047440	020116	
12847	101554	040526	044514	020104	
12848	101562	052123	051117	020105	
12849	101570	040520	044522	054524	
12850	101576	042440	051122	051117	
12851	101604	000			
12852	101605	124	051505	026524	EM104: .ASCII /TEST-DATA-REFERENCE GIVING VALID STORE PARITY/
12853	101612	040504	040524	051055	
12854	101620	043105	051105	047105	
12855	101626	042503	043440	053111	
12856	101634	047111	020107	040526	
12857	101642	044514	020104	052123	
12858	101650	051117	020105	040520	
12859	101656	044522	054524		
12860	101662	005015	051105	047522	.ASCIZ <15><12>/ERROR WAS NOT A MISS/
12861	101670	020122	040527	020123	
12862	101676	047516	020124	020101	
12863	101704	044515	051523	000	
12864	101711	106	050126	020105	EM105: .ASCIZ /FVPE DID NOT GET CLEARED AFTER VSPE OCCURED/
12865	101716	044504	020104	047516	
12866	101724	020124	042507	020124	
12867	101732	046103	040505	042522	
12868	101740	020104	043101	042524	
12869	101746	020122	051526	042520	
12870	101754	047440	041503	051125	
12871	101762	042105	000		
12872	101765	126	046101	042111	EM106: .ASCIZ /VALID-STORE-PARITY-ERROR BIT DID NOT SET IN CCR ON VSPE/
12873	101772	051455	047524	042522	
12874	102000	050055	051101	052111	
12875	102006	026531	051105	047522	
12876	102014	020122	044502	020124	
12877	102022	044504	020104	047516	
12878	102030	020124	042523	020124	
12879	102036	047111	041440	051103	
12880	102044	047440	020116	051526	
12881	102052	042520	000		
12882	102055	106	051501	020124	EM107: .ASCII /FAST ADDRESS MEMORY PARITY ERROR BITS (4,5) NOT/
12883	102062	042101	051104	051505	
12884	102070	020123	042515	047515	
12885	102076	054522	050040	051101	
12886	102104	052111	020131	051105	
12887	102112	047522	020122	044502	
12888	102120	051524	024040	026064	
12889	102126	024465	047040	052117	
12890	102134	005015	042523	020124	.ASCIZ <15><12>/SET CORRECTLY IN MSER ON VSPE/
12891	102142	047503	051122	041505	
12892	102150	046124	020131	047111	

12893	102156	046440	042523	020122	
12894	102164	047117	053040	050123	
12895	102172	000105			
12896	102174	051526	052511	051440	EM110: .ASCIZ /VSIU SWITCHED ON VSPE/
12897	102202	044527	041524	042510	
12898	102210	020104	047117	053040	
12899	102216	050123	000105		
12900	102222	042515	047515	054522	EM111: .ASCIZ /MEMORY SYSTEM ERROR REGISTER COULD NOT BE CLEARED/
12901	102230	051440	051531	042524	
12902	102236	020115	051105	047522	
12903	102244	020122	042522	044507	
12904	102252	052123	051105	041440	
12905	102260	052517	042114	047040	
12906	102266	052117	041040	020105	
12907	102274	046103	040505	042522	
12908	102302	000104			
12909	102304	051526	042520	041440	EM112: .ASCIZ /VSPE COULD NOT BE CLEARED IN CCR/
12910	102312	052517	042114	047040	
12911	102320	052117	041040	020105	
12912	102326	046103	040505	042522	
12913	102334	020104	047111	041440	
12914	102342	051103	000		
12915	102345	124	051505	026524	EM113: .ASCIZ /TEST-DATA-REFERENCE NOT A HIT/
12916	102352	040504	040524	051055	
12917	102360	043105	051105	047105	
12918	102366	042503	047040	052117	
12919	102374	040440	044040	052111	
12920	102402	000			
12921	102403	124	051505	026524	EM115: .ASCII /TEST-DATA-REFERNECE NOT A MISS/
12922	102410	040504	040524	051055	
12923	102416	043105	051105	042516	
12924	102424	042503	047040	052117	
12925	102432	040440	046440	051511	
12926	102440	123			
12927	102441	015	041412	041501	.ASCIZ <15><12>/CACHE DID NOT TURN OFF ON BACK-TO-BACK FLUSH/
12928	102446	042510	042040	042111	
12929	102454	047040	052117	052040	
12930	102462	051125	020116	043117	
12931	102470	020106	047117	041040	
12932	102476	041501	026513	047524	
12933	102504	041055	041501	020113	
12934	102512	046106	051525	000110	
12935					
12936	102520	054502	020120	044502	EM123: .ASCIZ ?BYP BIT IN KIPDR COULD NOT BE CLEARED?
12937	102526	020124	047111	045440	
12938	102534	050111	051104	041440	
12939	102542	052517	042114	047040	
12940	102550	052117	041040	020105	
12941	102556	046103	040505	042522	
12942	102564	000104			
12943	102566	054502	020120	044502	EM124: .ASCIZ ?BYP BIT IN KIPDR COULD NOT BE SET?
12944	102574	020124	047111	045440	
12945	102602	050111	051104	041440	
12946	102610	052517	042114	047040	
12947	102616	052117	041040	020105	
12948	102624	042523	000124		

12949	102630	042524	052123	042055
12950	102636	052101	020101	047503
12951	102644	046125	020104	047516
12952	102652	020124	042502	046440
12953	102660	042101	020105	044510
12954	102666	000124		
12955	102670	042524	052123	042055
12956	102676	052101	020101	042522
12957	102704	042506	042522	041516
12958	102712	020105	047516	020124
12959	102720	020101	044515	051523
12960	102726	005015	040503	044103
12961	102734	042105	042040	052101
12962	102742	020101	040527	020123
12963	102750	047516	020124	047506
12964	102756	041522	042105	040440
12965	102764	046440	051511	020123
12966	102772	047117	053040	051111
12967	103000	052524	046101	050040
12968	103006	043501	020105	054502
12969	103014	040520	051523	000
12970	103021	124	051505	020124
12971	103026	040504	040524	051040
12972	103034	043105	051105	047105
12973	103042	042503	047040	052117
12974	103050	040440	046440	051511
12975	103056	123		
12976	103057	015	041412	041501
12977	103064	042510	020104	040504
12978	103072	040524	053440	051501
12979	103100	047040	052117	044440
12980	103106	053116	046101	042111
12981	103114	052101	042105	047440
12982	103122	020116	044526	052122
12983	103130	040525	020114	040520
12984	103136	042507	041040	050131
12985	103144	051501	000123	
12986	103150	054502	020120	044502
12987	103156	020124	047111	051440
12988	103164	050111	051104	041440
12989	103172	052517	042114	047040
12990	103200	052117	041040	020105
12991	103206	046103	040505	042522
12992	103214	000104		
12993	103216	054502	020120	044502
12994	103224	020124	047111	051440
12995	103232	050111	051104	041440
12996	103240	052517	042114	047040
12997	103246	052117	041040	020105
12998	103254	042523	000124	
12999	103260	054502	020120	044502
13000	103266	020124	047111	052440
13001	103274	050111	051104	041440
13002	103302	052517	042114	047040
13003	103310	052117	041040	020105
13004	103316	046103	040505	042522

EM125: .ASCIZ /TEST-DATA COULD NOT BE MADE HIT/

EM126: .ASCII /TEST-DATA REFERENCE NOT A MISS/

.ASCIZ <15><12>/CACHED DATA WAS NOT FORCED A MISS ON VIRTUAL PAGE BYPASS/

EM127: .ASCII /TEST DATA REFERENCE NOT A MISS/

.ASCIZ <15><12>/CACHED DATA WAS NOT INVALIDATED ON VIRTUAL PAGE BYPASS/

EM130: .ASCIZ ?BYP BIT IN SIPDR COULD NOT BE CLEARED?

EM131: .ASCIZ ?BYP BIT IN SIPDR COULD NOT BE SET?

EM132: .ASCIZ ?BYP BIT IN UIPDR COULD NOT BE CLEARED?

13005	103324	000104				
13006	103326	054502	020120	044502	EM133:	.ASCIZ ?BYP BIT IN UIPDR COULD NOT BE SET?
13007	103334	020124	047111	052440		
13008	103342	050111	051104	041440		
13009	103350	052517	042114	047040		
13010	103356	052117	041040	020105		
13011	103364	042523	000124			
13012	103370	040503	044103	020105	EM136:	.ASCII 'CACHE ADDRESS MEMORY PARITY LOGIC TEST FAILED.' <crlf&gt;< td=""> </crlf&gt;<>
13013	103376	042101	051104	051505		
13014	103404	020123	042515	047515		
13015	103412	054522	050040	051101		
13016	103420	052111	020131	047514		
13017	103426	044507	020103	042524		
13018	103434	052123	043040	044501		
13019	103442	042514	027104	200		
13020	103447	125	040516	046102		.ASCII 'UNABLE TO FORCE A PARITY ERROR ON THE LOW BYTE '
13021	103454	020105	047524	043040		
13022	103462	051117	042503	040440		
13023	103470	050040	051101	052111		
13024	103476	020131	051105	047522		
13025	103504	020122	047117	052040		
13026	103512	042510	046040	053517		
13027	103520	041040	052131	020105		
13028	103526	043117	040440	020116		.ASCIZ 'OF AN ADDRESS.' <crlf&gt;'using maintenance="" register.'<="" td="" the=""> </crlf&gt;'using>
13029	103534	042101	051104	051505		
13030	103542	026123	052600	044523		
13031	103550	043516	052040	042510		
13032	103556	046440	044501	052116		
13033	103564	047105	047101	042503		
13034	103572	051040	043505	051511		
13035	103600	042524	027122	000		
13036						
13037	103605	103	041501	042510	EM137:	.ASCII 'CACHE ADDRESS MEMORY PARITY LOGIC TEST FAILED.'
13038	103612	040440	042104	042522		
13039	103620	051523	046440	046505		
13040	103626	051117	020131	040520		
13041	103634	044522	054524	046040		
13042	103642	043517	041511	052040		
13043	103650	051505	020124	040506		
13044	103656	046111	042105	056		
13045	103663	200	047125	041101		.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR ON THE HIGH BYTE '
13046	103670	042514	052040	020117		
13047	103676	047506	041522	020105		
13048	103704	020101	040520	044522		
13049	103712	054524	042440	051122		
13050	103720	051117	047440	020116		
13051	103726	044124	020105	044510		
13052	103734	044107	041040	052131		
13053	103742	020105				
13054	103744	043117	040440	020116		.ASCIZ 'OF AN ADDRESS.' <crlf&gt;'using maintenance="" register.'<="" td="" the=""> </crlf&gt;'using>
13055	103752	042101	051104	051505		
13056	103760	026123	052600	044523		
13057	103766	043516	052040	042510		
13058	103774	046440	044501	052116		
13059	104002	047105	047101	042503		
13060	104010	051040	043505	051511		

13061	104016	042524	027122	000
13062				
13063	104023			
13064	104023	115	044501	020116
13065	104030	042515	047515	054522
13066	104036	042040	052101	020101
13067	104044	040520	044522	054524
13068	104052	041440	042510	045503
13069	104060	051105	020123	042524
13070	104066	052123	043040	044501
13071	104074	042514	027104	
13072	104100	052600	040516	046102
13073	104106	020105	047524	043040
13074	104114	051117	042503	040440
13075	104122	050040	051101	052111
13076	104130	020131	051105	047522
13077	104136	026122	052440	044523
13078	104144	043516	040	
13079	104147	124	042510	046440
13080	104154	044501	052116	047105
13081	104162	047101	042503	051040
13082	104170	043505	051511	042524
13083	104176	026122	200	
13084	104201	101	020124	044124
13085	104206	020105	040515	047111
13086	104214	046440	046505	051117
13087	104222	020131	053105	047105
13088	104230	053440	051117	026104
13089	104236	046040	053517	041040
13090	104244	052131	026105	050040
13091	104252	051101	052111	020131
13092	104260	044103	041505	042513
13093	104266	026122	020200	042522
13094	104274	042101	047111	020107
13095	104302	020101	040504	040524
13096	104310	050040	052101	042524
13097	104316	047122	053440	044510
13098	104324	044103	040	
13099	104327	123	047510	046125
13100	104334	020104	040510	042526
13101	104342	041440	052501	042523
13102	104350	020104	047101	042440
13103	104356	051122	051117	000056
13104				
13105	104364			
13106	104364	040515	047111	046440
13107	104372	046505	051117	020131
13108	104400	040504	040524	050040
13109	104406	051101	052111	020131
13110	104414	044103	041505	042513
13111	104422	051522	052040	051505
13112	104430	020124	040506	046111
13113	104436	042105	056	
13114	104441	200	047125	041101
13115	104446	042514	052040	020117
13116	104454	047506	041522	020105

EM140: .ASCII 'MAIN MEMORY DATA PARITY CHECKERS TEST FAILED.'

.ASCII <CRLF> 'UNABLE TO FORCE A PARITY ERROR, USING '

.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>

.ASCII 'AT THE MAIN MEMORY EVEN WORD, LOW BYTE, PARITY '

.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '

.ASCII 'SHOULD HAVE CAUSED AN ERROR.'

EM141: .ASCII 'MAIN MEMORY DATA PARITY CHECKERS TEST FAILED.'

.ASCII <CRLF> 'UNABLE TO FORCE A PARITY ERROR, USING '



13117	104462	020101	040520	044522	
13118	104470	054524	042440	051122	
13119	104476	051117	020054	051525	
13120	104504	047111	020107		
13121	104510	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
13122	104516	047111	042524	040516	
13123	104524	041516	020105	042522	
13124	104532	044507	052123	051105	
13125	104540	100054			
13126	104542	052101	052040	042510	.ASCII 'AT THE MAIN MEMORY ODD WORD, LOW BYTE, PARITY '
13127	104550	046440	044501	020116	
13128	104556	042515	047515	054522	
13129	104564	047440	042104	053440	
13130	104572	051117	026104	046040	
13131	104600	053517	041040	052131	
13132	104606	026105	050040	051101	
13133	104614	052111	020131		
13134	104620	044103	041505	042513	.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '
13135	104626	026122	020200	042522	
13136	104634	042101	047111	020107	
13137	104642	020101	040504	040524	
13138	104650	050040	052101	042524	
13139	104656	047122	053440	044510	
13140	104664	044103	040		
13141	104667	123	047510	046125	.ASCIIZ 'SHOULD HAVE CAUSED AN ERROR.'
13142	104674	020104	040510	042526	
13143	104702	041440	052501	042523	
13144	104710	020104	047101	042440	
13145	104716	051122	051117	000056	
13146					
13147	104724				EM142:
13148	104724	040515	047111	046440	.ASCII 'MAIN MEMORY DATA PARITY CHECKERS TEST FAILED.'
13149	104732	046505	051117	020131	
13150	104740	040504	040524	050040	
13151	104746	051101	052111	020131	
13152	104754	044103	041505	042513	
13153	104762	051522	052040	051505	
13154	104770	020124	040506	046111	
13155	104776	042105	056		
13156	105001	200	047125	041101	.ASCII <CRLF> 'UNABLE TO FORCE A PARITY ERROR, USING '
13157	105006	042514	052040	020117	
13158	105014	047506	041522	020105	
13159	105022	020101	040520	044522	
13160	105030	054524	042440	051122	
13161	105036	051117	020054	051525	
13162	105044	047111	020107		
13163	105050	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
13164	105056	047111	042524	040516	
13165	105064	041516	020105	042522	
13166	105072	044507	052123	051105	
13167	105100	100054			
13168	105102	052101	052040	042510	.ASCII 'AT THE MAIN MEMORY EVEN WORD, HIGH BYTE, PARITY '
13169	105110	046440	044501	020116	
13170	105116	042515	047515	054522	
13171	105124	042440	042526	020116	
13172	105132	047527	042122	020054	

13173	105140	044510	044107	041040	
13174	105146	052131	026105	050040	
13175	105154	051101	052111	020131	
13176	105162	044103	041505	042513	.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '
13177	105170	026122	020200	042522	
13178	105176	042101	047111	020107	
13179	105204	020101	040504	040524	
13180	105212	050040	052101	042524	
13181	105220	047122	053440	044510	
13182	105226	044103	040		
13183	105231	123	047510	046125	.ASCIZ 'SHOULD HAVE CAUSED AN ERROR.'
13184	105236	020104	040510	042526	
13185	105244	041440	052501	042523	
13186	105252	020104	047101	042440	
13187	105260	051122	051117	000056	
13188					
13189	105266				EM143:
13190	105266	040515	047111	046440	.ASCII 'MAIN MEMORY DATA PARITY CHECKERS TEST FAILED.'
13191	105274	046505	051117	020131	
13192	105302	040504	040524	050040	
13193	105310	051101	052111	020131	
13194	105316	044103	041505	042513	
13195	105324	051522	052040	051505	
13196	105332	020124	040506	046111	
13197	105340	042105	056		
13198	105343	200	047125	041101	.ASCII <CRLF> 'UNABLE TO FORCE A PARITY ERROR, USING '
13199	105350	042514	052040	020117	
13200	105356	047506	041522	020105	
13201	105364	020101	040520	044522	
13202	105372	054524	042440	051122	
13203	105400	051117	020054	051525	
13204	105406	047111	020107		
13205	105412	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
13206	105420	047111	042524	040516	
13207	105426	041516	020105	042522	
13208	105434	044507	052123	051105	
13209	105442	100054			
13210	105444	052101	052040	042510	.ASCII 'AT THE MAIN MEMORY ODD WORD, HIGH BYTE, PARITY '
13211	105452	046440	044501	020116	
13212	105460	042515	047515	054522	
13213	105466	047440	042104	053440	
13214	105474	051117	026104	044040	
13215	105502	043511	020110	054502	
13216	105510	042524	020054	040520	
13217	105516	044522	054524	040	
13218	105523	103	042510	045503	.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '
13219	105530	051105	100054	051040	
13220	105536	040505	044504	043516	
13221	105544	040440	042040	052101	
13222	105552	020101	040520	052124	
13223	105560	051105	020116	044127	
13224	105566	041511	020110		
13225	105572	044123	052517	042114	.ASCIZ 'SHOULD HAVE CAUSED AN ERROR.'
13226	105600	044040	053101	020105	
13227	105606	040503	051525	042105	
13228	105614	040440	020116	051105	

13229	105622	047522	027122	000	
13230					
13231	105627				EM144:
13232	105627	103	041501	042510	.ASCII 'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'
13233	105634	042040	052101	020101	
13234	105642	042515	047515	054522	
13235	105650	050040	051101	052111	
13236	105656	020131	044103	041505	
13237	105664	042513	051522	052040	
13238	105672	051505	020124	040506	
13239	105700	046111	042105	056	
13240	105705	200	047125	041101	.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
13241	105712	042514	052040	020117	
13242	105720	047506	041522	020105	
13243	105726	020101	040520	044522	
13244	105734	054524	042440	051122	
13245	105742	051117	020054	051525	
13246	105750	047111	020107		
13247	105754	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
13248	105762	047111	042524	040516	
13249	105770	041516	020105	042522	
13250	105776	044507	052123	051105	
13251	106004	100054			
13252	106006	052101	052040	042510	.ASCII 'AT THE GROUP ZERO,LOW BYTE, DATA PARITY CHECKER,'
13253	106014	043440	047522	050125	
13254	106022	055040	051105	026117	
13255	106030	047514	020127	054502	
13256	106036	042524	020054	040504	
13257	106044	040524	050040	051101	
13258	106052	052111	020131	044103	
13259	106060	041505	042513	026122	
13260	106066	051200	040505	044504	.ASCII <CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
13261	106074	043516	040440	042040	
13262	106102	052101	020101	040520	
13263	106110	052124	051105	020116	
13264	106116	044127	041511	020110	
13265	106124	044123	052517	042114	
13266	106132	044040	053101	020105	
13267	106140	040503	051525	042105	.ASCIIZ 'CAUSED AN ERROR.'
13268	106146	040440	020116	051105	
13269	106154	047522	027122	000	
13270					
13271	106161				EM145:
13272	106161	103	041501	042510	.ASCII 'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'
13273	106166	042040	052101	020101	
13274	106174	042515	047515	054522	
13275	106202	050040	051101	052111	
13276	106210	020131	044103	041505	
13277	106216	042513	051522	052040	
13278	106224	051505	020124	040506	
13279	106232	046111	042105	056	
13280	106237	200	047125	041101	.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
13281	106244	042514	052040	020117	
13282	106252	047506	041522	020105	
13283	106260	020101	040520	044522	
13284	106266	054524	042440	051122	

13285	106274	051117	020054	051525	
13286	106302	047111	020107		
13287	106306	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,<CRLF>
13288	106314	047111	042524	040516	
13289	106322	041516	020105	042522	
13290	106330	044507	052123	051105	
13291	106336	100054			
13292	106340	052101	052040	042510	.ASCII 'AT THE GROUP ONE,LOW BYTE, DATA PARITY CHECKER,'
13293	106346	043440	047522	050125	
13294	106354	047440	042516	046054	
13295	106362	053517	041040	052131	
13296	106370	026105	042040	052101	
13297	106376	020101	040520	044522	
13298	106404	054524	041440	042510	
13299	106412	045503	051105	054	
13300	106417	200	042522	042101	.ASCII <CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
13301	106424	047111	020107	020101	
13302	106432	040504	040524	050040	
13303	106440	052101	042524	047122	
13304	106446	053440	044510	044103	
13305	106454	051440	047510	046125	
13306	106462	020104	040510	042526	
13307	106470	040			
13308	106471	103	052501	042523	.ASCIZ 'CAUSED AN ERROR.'
13309	106476	020104	047101	042440	
13310	106504	051122	051117	000056	
13311					
13312	106512				EM146:
13313	106512	040503	044103	020105	.ASCII 'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'
13314	106520	040504	040524	046440	
13315	106526	046505	051117	020131	
13316	106534	040520	044522	054524	
13317	106542	041440	042510	045503	
13318	106550	051105	020123	042524	
13319	106556	052123	043040	044501	
13320	106564	042514	027104		
13321	106570	052600	040516	046102	.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
13322	106576	020105	047524	043040	
13323	106604	051117	042503	040440	
13324	106612	050040	051101	052111	
13325	106620	020131	051105	047522	
13326	106626	026122	052440	044523	
13327	106634	043516	040		
13328	106637	124	042510	046440	.ASCII 'THE MAINTENANCE REGISTER,<CRLF>
13329	106644	044501	052116	047105	
13330	106652	047101	042503	051040	
13331	106660	043505	051511	042524	
13332	106666	026122	200		
13333	106671	101	020124	044124	.ASCII 'AT THE GROUP ZERO,HIGH BYTE, DATA PARITY CHECKER,'
13334	106676	020105	051107	052517	
13335	106704	020120	042532	047522	
13336	106712	044054	043511	020110	
13337	106720	054502	042524	020054	
13338	106726	040504	040524	050040	
13339	106734	051101	052111	020131	
13340	106742	044103	041505	042513	

13341	106750	026122				
13342	106752	051200	040505	044504	.ASCII	<CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
13343	106760	043516	040440	042040		
13344	106766	052101	020101	040520		
13345	106774	052124	051105	020116		
13346	107002	044127	041511	020110		
13347	107010	044123	052517	042114		
13348	107016	044040	053101	020105		
13349	107024	040503	051525	042105	.ASCIZ	'CAUSED AN ERROR.'
13350	107032	040440	020116	051105		
13351	107040	047522	027122	000		
13352						
13353	107045				EM147:	
13354	107045	103	041501	042510	.ASCII	'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'
13355	107052	042040	052101	020101		
13356	107060	042515	047515	054522		
13357	107066	050040	051101	052111		
13358	107074	020131	044103	041505		
13359	107102	042513	051522	052040		
13360	107110	051505	020124	040506		
13361	107116	046111	042105	056		
13362	107123	200	047125	041101	.ASCII	<CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
13363	107130	042514	052040	020117		
13364	107136	047506	041522	020105		
13365	107144	020101	040520	044522		
13366	107152	054524	042440	051122		
13367	107160	051117	020054	051525		
13368	107166	047111	020107			
13369	107172	044124	020105	040515	.ASCII	'THE MAINTENANCE REGISTER,'<CRLF>
13370	107200	047111	042524	040516		
13371	107206	041516	020105	042522		
13372	107214	044507	052123	051105		
13373	107222	100054				
13374	107224	052101	052040	042510	.ASCII	'AT THE GROUP ONE,HIGH BYTE, DATA PARITY CHECKER,'
13375	107232	043440	047522	050125		
13376	107240	047440	042516	044054		
13377	107246	043511	020110	054502		
13378	107254	042524	020054	040504		
13379	107262	040524	050040	051101		
13380	107270	052111	020131	044103		
13381	107276	041505	042513	026122		
13382	107304	051200	040505	044504	.ASCII	<CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
13383	107312	043516	040440	042040		
13384	107320	052101	020101	040520		
13385	107326	052124	051105	020116		
13386	107334	044127	041511	020110		
13387	107342	044123	052517	042114		
13388	107350	044040	053101	020105		
13389	107356	040503	051525	042105	.ASCIZ	'CAUSED AN ERROR.'
13390	107364	040440	020116	051105		
13391	107372	047522	027122	000		
13392						
13393	107377	200	047125	054105	EM150:	.ASCIZ <CRLF>'UNEXPECTED CPU ERROR TRAPPED TO VECTOR ERRVEC (4)!'
13394	107404	042520	052103	042105		
13395	107412	041440	052520	042440		
13396	107420	051122	051117	052040		

13397	107426	040522	050120	042105	
13398	107434	052040	020117	042526	
13399	107442	052103	051117	042440	
13400	107450	051122	042526	020103	
13401	107456	032050	020451	000	
13402					
13403	107463	115	051501	020123	EM151: .ASCIZ 'MASS BUS WRITE HIT DID NOT INVALIDATE THE CACHE.'
13404	107470	052502	020123	051127	
13405	107476	052111	020105	044510	
13406	107504	020124	044504	020104	
13407	107512	047516	020124	047111	
13408	107520	040526	044514	040504	
13409	107526	042524	052040	042510	
13410	107534	041440	041501	042510	
13411	107542	000056			
13412					
13413		107463			EM152=EM151
13414		107463			EM153=EM151
13415					
13416	107544	042504	044526	042503	EM154: .ASCIZ 'DEVICE ERROR IN THE RS04.'
13417	107552	042440	051122	051117	
13418	107560	044440	020116	044124	
13419	107566	020105	051522	032060	
13420	107574	000056			
13421					
13422	107576	042504	044526	042503	EM155: .ASCIZ 'DEVICE ERROR IN THE RP04.'
13423	107604	042440	051122	051117	
13424	107612	044440	020116	044124	
13425	107620	020105	050122	032060	
13426	107626	000056			
13427					
13428	107630	042504	044526	042503	EM156: .ASCIZ 'DEVICE ERROR IN THE MASS BUS TESTER.'
13429	107636	042440	051122	051117	
13430	107644	044440	020116	044124	
13431	107652	020105	040515	051523	
13432	107660	041040	051525	052040	
13433	107666	051505	042524	027122	
13434	107674	000			
13435					
13436					
13437	107675	104	053105	041511	EM160: .ASCIZ 'DEVICE ERROR IN THE RK05.'
13438	107702	020105	051105	047522	
13439	107710	020122	047111	052040	
13440	107716	042510	051040	030113	
13441	107724	027065	000		
13442					
13443	107727	104	053105	041511	EM161: .ASCIZ 'DEVICE ERROR IN THE UNIBUS EXERCISER.'
13444	107734	020105	051105	047522	
13445	107742	020122	047111	052040	
13446	107750	042510	052440	044516	
13447	107756	052502	020123	054105	
13448	107764	051105	044503	042523	
13449	107772	027122	000		
13450	107775	125	041502	020102	EM162: .ASCII /UBCB PE ABORT DOESN'T GO LOW OR/<CRLF>
13451	110002	042520	040440	047502	
13452	110010	052122	042040	042517	

13453	110016	047123	052047	043440
13454	110024	020117	047514	020127
13455	110032	051117	200	
13456	110035	111	020124	047504
13457	110042	051505	023516	020124
13458	110050	042507	020124	047524
13459	110056	052040	041515	020103
13460	110064	031505	020063	051117
13461	110072	042440	031463	041040
13462	110100	042101	200	
13463	110103	117	020122	041125
13464	110110	041103	050040	051101
13465	110116	052111	020131	051105
13466	110124	020122	047504	051505
13467	110132	023516	020124	042507
13468	110140	020124	047524	052040
13469	110146	041515	020102	032505
13470	110154	100063		
13471	110156	051501	040440	046040
13472	110164	053517	047440	020122
13473	110172	032505	024063	024465
13474	110200	041040	042101	000
13475	110205	125	041502	020102
13476	110212	040520	044522	054524
13477	110220	042440	051122	042040
13478	110226	042517	047123	052047
13479	110234	043440	020117	047514
13480	110242	020127	051117	044440
13481	110250	020124	047504	051505
13482	110256	200		
13483	110257	116	052117	043440
13484	110264	052105	052040	020117
13485	110272	040504	042520	000
13486	110277	104	050101	020105
13487	110304	030505	024061	024464
13488	110312	041040	042101	047440
13489	110320	020122	053124	033060
13490	110326	042040	042517	047123
13491	110334	052047	043440	052105
13492	110342	052040	020117	044124
13493	110350	020105	046101	000125
13494	110356	040504	042520	042440
13495	110364	024067	024461	041040
13496	110372	042101	000	
13497	110375	124	041515	020101
13498	110402	042523	025507	047503
13499	110410	025516	040520	020122
13500	110416	047504	051505	023516
13501	110424	020124	047507	046040
13502	110432	053517	047440	020116
13503	110440	041503	045102	050040
13504	110446	051101	052111	020131
13505	110454	051124	050101	000
13506	110461	124	041515	020102
13507	110466	040520	052122	042040
13508	110474	042517	047123	052047

.ASCII /IT DOESN'T GET TO TMCC E33 OR E33 BAD/<CRLF>

.ASCII /OR UBCB PARITY ERR DOESN'T GET TO TMCB E53/<CRLF>

.ASCIZ /AS A LOW OR E53(5) BAD/

EM163: .ASCII /UBCB PARITY ERR DOESN'T GO LOW OR IT DOES/<CRLF>

.ASCIZ /NOT GET TO DAPE/

EM164: .ASCIZ /DAPE E11(4) BAD OR TV06 DOESN'T GET TO THE ALU/

EM165: .ASCIZ /DAPE E7(1) BAD/

EM166: .ASCIZ /TMCA SEG+CON+PAR DOESN'T GO LOW ON CCBJ PARITY TRAP/

EM167: .ASCII /TMCB PART DOESN'T GO LOW OR DOES/<CRLF>

13509	110502	043440	020117	047514	
13510	110510	020127	051117	042040	
13511	110516	042517	100123		
13512	110522	047516	020124	042507	.ASCIZ /NOT GET TO UBCB OR UBCB E18(1) BAD/
13513	110530	020124	047524	052440	
13514	110536	041502	020102	051117	
13515	110544	052440	041502	020102	
13516	110552	030505	024070	024461	
13517	110560	041040	042101	000	
13518	110565	124	041515	020101	EM170: .ASCIZ /TMCA E67(8) DOESN'T GO LOW ON MGMT/
13519	110572	033105	024067	024470	
13520	110600	042040	042517	047123	
13521	110606	052047	043440	020117	
13522	110614	047514	020127	047117	
13523	110622	046440	046507	000124	
13524	110630	046524	040503	042440	EM171: .ASCIZ /TMCA E67(12) DOESN'T GO LOW ON MGMT/
13525	110636	033466	030450	024462	
13526	110644	042040	042517	047123	
13527	110652	052047	043440	020117	
13528	110660	047514	020127	047117	
13529	110666	046440	046507	000124	
13530	110674	046524	040503	042440	EM172: .ASCII /TMCA E68(6) DOESN'T GO LOW ON PAR TRP/<CRLF>
13531	110702	034066	033050	020051	
13532	110710	047504	051505	023516	
13533	110716	020124	047507	046040	
13534	110724	053517	047440	020116	
13535	110732	040520	020122	051124	
13536	110740	100120			
13537	110742	051117	042440	032464	.ASCIZ /OR E45(4) BAD/
13538	110750	032050	020051	040502	
13539	110756	000104			
13540	110760	046524	040503	042440	EM173: .ASCIZ /TMCA E68(8) DOESN'T GO LOW ON PAR TRP/
13541	110766	034066	034050	020051	
13542	110774	047504	051505	023516	
13543	111002	020124	047507	046040	
13544	111010	053517	047440	020116	
13545	111016	040520	020122	051124	
13546	111024	000120			
13547	111026	046524	041503	050040	EM435: .ASCII /TMCC PRIORITY CLEAR DIDN'T GO LOW OR DIDN'T/<CRLF>
13548	111034	044522	051117	052111	
13549	111042	020131	046103	040505	
13550	111050	020122	044504	047104	
13551	111056	052047	043440	020117	
13552	111064	047514	020127	051117	
13553	111072	042040	042111	023516	
13554	111100	100124			
13555	111102	042507	020124	044124	.ASCIZ /GET THRU TMCA E43(2) ON ABORT CLEAR/
13556	111110	052522	052040	041515	
13557	111116	020101	032105	024063	
13558	111124	024462	047440	020116	
13559	111132	041101	051117	020124	
13560	111140	046103	040505	000122	
13561	111146	052502	020123	041120	EM174: .ASCIZ /BUS PB DIDN'T GET TO UBCB PE ABORT/
13562	111154	042040	042111	023516	
13563	111162	020124	042507	020124	
13564	111170	047524	052440	041502	



13565	111176	020102	042520	040440	
13566	111204	047502	052122	000	
13567	111211	125	041502	020102	EM175: .ASCIZ /UBCB PARITY ERR DIDN'T GO LOW ON BUS PB/
13568	111216	040520	044522	054524	
13569	111224	042440	051122	042040	
13570	111232	042111	023516	020124	
13571	111240	047507	046040	053517	
13572	111246	047440	020116	052502	
13573	111254	020123	041120	000	
13574					
13575					;THESE ARE DATA HEADERS:
13576					
13577	111261	040	052040	051505	DH1: .ASCIZ ' TEST.' <tab>' GROUP.'<tab>'PHYSICAL ADDR.'<tab>'CALL AT PC.'</tab></tab></tab>
13578	111266	027124	020011	051107	
13579	111274	052517	027120	050011	
13580	111302	054510	044523	040503	
13581	111310	020114	042101	051104	
13582	111316	004456	040503	046114	
13583	111324	040440	020124	041520	
13584	111332	000056			
13585					
13586	111334	020040	042524	052123	DH2: .ASCII ' TEST.' <tab>' GROUP.'<tab>'ERROR ADDR REG.'<tab>'ERROR REG.'<tab></tab></tab></tab></tab>
13587	111342	004456	043440	047522	
13588	111350	050125	004456	051105	
13589	111356	047522	020122	042101	
13590	111364	051104	051040	043505	
13591	111372	004456	051105	047522	
13592	111400	020122	042522	027107	
13593	111406	011			
13594	111407	122	043105	040440	.ASCIZ 'REF ADDR.' <tab>'TRAP AT PC.'</tab>
13595	111414	042104	027122	052011	
13596	111422	040522	020120	052101	
13597	111430	050040	027103	000	
13598					
13599		111334			DH3=DH2
13600					
13601		111334			DH4=DH2
13602					
13603	111435	040	052040	051505	DH5: .ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>'READ.'</tab></tab>
13604	111442	027124	041411	046101	
13605	111450	020114	052101	050040	
13606	111456	027103	051011	040505	
13607	111464	027104	000		
13608					
13609		111435			DH6=DH5
13610					
13611		111435			DH7=DH5
13612					
13613		111435			DH10=DH5
13614					
13615	111467	040	052040	051505	DH11: .ASCIZ ' TEST.' <tab>' GROUP.'<tab>'TRAP AT PC.'<tab>'ERROR ADDR REG.'</tab></tab></tab>
13616	111474	027124	020011	051107	
13617	111502	052517	027120	052011	
13618	111510	040522	020120	052101	
13619	111516	050040	027103	042411	
13620	111524	051122	051117	040440	

13621	111532	042104	020122	042522		
13622	111540	027107	000			
13623						
13624	111543	040	052040	051505	DH12:	.ASCII ' TEST.<TAB>' GROUP.<TAB>'CALL AT PC.<TAB>'TEST ADDR.<TAB>
13625	111550	027124	020011	051107		
13626	111556	052517	027120	041411		
13627	111564	046101	020114	052101		
13628	111572	050040	027103	052011		
13629	111600	051505	020124	042101		
13630	111606	051104	004456			
13631	111612	040504	040524	053440		.ASCIZ 'DATA WR. DATA READ.'
13632	111620	027122	042040	052101		
13633	111626	020101	042522	042101		
13634	111634	000056				
13635						
13636	111636	020040	042524	052123	DH13:	.ASCII ' TEST.<TAB>' GROUP.<TAB>'*DATA.<TAB>'+DATA.<TAB>
13637	111644	004456	043440	047522		
13638	111652	050125	004456	042052		
13639	111660	052101	027101	025411		
13640	111666	040504	040524	004456		
13641	111674	051105	047522	020122		.ASCIZ 'ERROR COUNT.'
13642	111702	047503	047125	027124		
13643	111710	000				
13644						
13645	111711	040	052040	051505	DH14:	.ASCII ' TEST.<TAB>'CALL AT PC.<TAB>'ERROR ADDR REG.'
13646	111716	027124	041411	046101		
13647	111724	020114	052101	050040		
13648	111732	027103	042411	051122		
13649	111740	051117	040440	042104		
13650	111746	020122	042522	027107		
13651	111754	052011	040522	020120		.ASCII <TAB>'TRAP AT PC.<TAB>
13652	111762	052101	050040	027103		
13653	111770	011				
13654	111771	105	051122	051117		.ASCIZ 'ERROR REG.'
13655	111776	051040	043505	000056		
13656						
13657	112004	020040	042524	052123	DH15:	.ASCIZ ' TEST.<TAB>'CALL AT PC.'
13658	112012	004456	040503	046114		
13659	112020	040440	020124	041520		
13660	112026	000056				
13661						
13662	112030	020040	042524	052123	DH16:	.ASCII ' TEST.<TAB>' GROUP.<TAB>'WROTE.<TAB>'READ.<TAB>
13663	112036	004456	043440	047522		
13664	112044	050125	004456	051127		
13665	112052	052117	027105	051011		
13666	112060	040505	027104	011		
13667	112065	101	042104	020122		.ASCIZ 'ADDR TESTED.<TAB>'CALL AT PC.'
13668	112072	042524	052123	042105		
13669	112100	004456	040503	046114		
13670	112106	040440	020124	041520		
13671	112114	000056				
13672						
13673	112116	020040	042524	052123	DH17:	.ASCII ' TEST.<TAB>' GROUP.<TAB>'ERROR AT PC.<TAB>'READ.<TAB>
13674	112124	004456	043440	047522		
13675	112132	050125	004456	051105		
13676	112140	047522	020122	052101		

13677	112146	050040	027103	051011	
13678	112154	040505	027104	011	
13679	112161	111	027116	040411	.ASCIZ 'IN.' <tab>'ADDRESS.'</tab>
13680	112166	042104	042522	051523	
13681	112174	000056			
13682					
13683		112116			DH20=DH17
13684					
13685	112176	020040	042524	052123	DH21: .ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>'READ.'<tab>' GROUP.'<tab>'ADDRESS.'</tab></tab></tab></tab>
13686	112204	004456	040503	046114	
13687	112212	040440	020124	041520	
13688	112220	004456	042522	042101	
13689	112226	004456	043440	047522	
13690	112234	050125	004456	042101	
13691	112242	051104	051505	027123	
13692	112250	000			
13693					
13694	112251	040	052040	051505	DH22: .ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>'EXPECTED ERROR AT.'</tab></tab>
13695	112256	027124	041411	046101	
13696	112264	020114	052101	050040	
13697	112272	027103	042411	050130	
13698	112300	041505	042524	020104	
13699	112306	051105	047522	020122	
13700	112314	052101	000056		
13701					
13702	112320	020040	042524	052123	DH23: .ASCII ' TEST.' <tab>'CALL AT PC.'<tab>'EXPECTED ADRS.'<tab></tab></tab></tab>
13703	112326	004456	040503	046114	
13704	112334	040440	020124	041520	
13705	112342	004456	054105	042520	
13706	112350	052103	042105	040440	
13707	112356	051104	027123	011	
13708	112363	107	052117	040440	.ASCIZ 'GOT ADRS.' <tab>'ERROR REG.'</tab>
13709	112370	051104	027123	042411	
13710	112376	051122	051117	051040	
13711	112404	043505	000056		
13712					
13713		112251			DH24=DH22
13714					
13715		112320			DH25=DH23
13716					
13717	112410	020040	042524	052123	DH26: .ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>' GROUP.'<tab>'ADDRESS.'</tab></tab></tab>
13718	112416	004456	040503	046114	
13719	112424	040440	020124	041520	
13720	112432	004456	043440	047522	
13721	112440	050125	004456	042101	
13722	112446	051104	051505	027123	
13723	112454	000			
13724					
13725	112455	040	052040	051505	DH27: .ASCII ' TEST.' <tab>'CALL AT PC.'<tab>' GROUP.'<tab>'ESTABLISHED HIT.'</tab></tab></tab>
13726	112462	027124	041411	046101	
13727	112470	020114	052101	050040	
13728	112476	027103	020011	051107	
13729	112504	052517	027120	042411	
13730	112512	052123	041101	044514	
13731	112520	044123	042105	044040	
13732	112526	052111	056		

```

13733 112531 040 052502 020124 .ASCIZ ' BUT GOT HIT.'
13734 112536 047507 020124 044510
13735 112544 027124 000
13736
13737 112251 DH30=DH22
13738
13739 112320 DH31=DH23
13740
13741 112251 DH32=DH22
13742
13743 112320 DH33=DH23
13744
13745 112547 040 052040 051505 DH34: .ASCII ' TEST.<TAB>'PC OF CALL.<TAB>'READ.<TAB>'IN ADDRESS.<TAB>'
13746 112554 027124 050011 020103
13747 112562 043117 041440 046101
13748 112570 027114 051011 040505
13749 112576 027104 044411 020116
13750 112604 042101 051104 051505
13751 112612 027123 011
13752 112615 105 050130 041505 .ASCIZ 'EXPECTED.'
13753 112622 042524 027104 000
13754
13755 112547 DH35=DH34
13756
13757 112627 040 052040 051505 DH36: .ASCIZ ' TEST.<TAB>'CALL AT PC.<TAB>' GROUP.<TAB>'ADDRESS.'
13758 112634 027124 041411 046101
13759 112642 020114 052101 050040
13760 112650 027103 020011 051107
13761 112656 052517 027120 040411
13762 112664 042104 042522 051523
13763 112672 000056
13764
13765 112674 020040 042524 052123 DH37: .ASCII ' TEST.<TAB>' GROUP.<TAB>'ERROR COUNT.<TAB>'
13766 112702 004456 043440 047522
13767 112710 050125 004456 051105
13768 112716 047522 020122 047503
13769 112724 047125 027124 011
13770 112731 052 041040 042101 .ASCIZ '* BAD ADRS.<TAB>' + BAD ADRS.'
13771 112736 040440 051104 027123
13772 112744 025411 041040 042101
13773 112752 040440 051104 027123
13774 112760 000
13775
13776
13777 112761 040 052040 051505 DH41: .ASCIZ ' TEST.<TAB>'CALL AT PC.<TAB>' GROUP.<TAB>'ADDRESS.'
13778 112766 027124 041411 046101
13779 112774 020114 052101 050040
13780 113002 027103 020011 051107
13781 113010 052517 027120 040411
13782 113016 042104 042522 051523
13783 113024 000056
13784
13785 112761 DH42=DH41
13786
13787 113026 020040 042524 052123 DH43: .ASCII ' TEST.<TAB>'CALL AT PC.<TAB>'TRAP AT PC.<TAB>' GROUP.'
13788 113034 004456 040503 046114

```

13789	113042	040440	020124	041520	
13790	113050	004456	051124	050101	
13791	113056	040440	020124	041520	
13792	113064	004456	043440	047522	
13793	113072	050125	056		
13794					
13795	113026				DH40=DH43
13796					
13797	113075	040	052040	051505	DH44: .ASCII ' TEST.' <tab>'CALL AT PC.'<tab>'TRAP AT PC.'<tab></tab></tab></tab>
13798	113102	027124	041411	046101	
13799	113110	020114	052101	050040	
13800	113116	027103	052011	040522	
13801	113124	020120	052101	050040	
13802	113132	027103	011		
13803	113135	105	051122	051117	.ASCIZ 'ERROR ADRS REG.' <tab>'ERROR REG.'</tab>
13804	113142	040440	051104	020123	
13805	113150	042522	027107	042411	
13806	113156	051122	051117	051040	
13807	113164	043505	000056		
13808					
13809	113075				DH45=DH44
13810					
13811	113170	020040	042524	052123	DH46: .ASCIZ ' TEST.' <tab>'CALL AT PC.'</tab>
13812	113176	004456	040503	046114	
13813	113204	040440	020124	041520	
13814	113212	000056			
13815					
13816	113170				DH47=DH46
13817					
13818	113075				DH50=DH44
13819					
13820	113075				DH51=DH44
13821					
13822	113170				DH52=DH46
13823					
13824	113170				DH53=DH46
13825					
13826	113214	020040	042524	052123	DH54: .ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>'ERROR COUNT.'</tab></tab>
13827	113222	004456	040503	046114	
13828	113230	040440	020124	041520	
13829	113236	004456	051105	047522	
13830	113244	020122	047503	047125	
13831	113252	027124	000		
13832	113255	040	050040	020103	DH55: .ASCIZ / PC CCR/
13833	113262	020040	041440	051103	
13834	113270	000			
13835	113271	040	050040	020103	DH66: .ASCIZ / PC CCR GROUP TST-DATA-ADRS/
13836	113276	020040	041440	051103	
13837	113304	020040	043440	047522	
13838	113312	050125	020040	052040	
13839	113320	052123	042055	052101	
13840	113326	026501	042101	051522	
13841	113334	000			
13842	113335	040	050040	020103	DH71: .ASCIZ / PC EXPCTD RECVD LOC/
13843	113342	020040	054105	041520	
13844	113350	042124	020040	042522	

13845	113356	053103	020104	046040								
13846	113364	041517	000									
13847	113367	040	050040	020103	DH107:	.ASCIZ / PC	CCR	MSER	GROUP	EXPCTD-B4,5/		
13848	113374	020040	041440	051103								
13849	113402	020040	020040	051515								
13850	113410	051105	020040	043440								
13851	113416	047522	050125	020040								
13852	113424	042440	050130	052103								
13853	113432	026504	032102	032454								
13854	113440	000										
13855	113441	040	050040	020103	DH111:	.ASCIZ / PC		MSER/				
13856	113446	020040	046440	042523								
13857	113454	000122										
13858	113456	020040	041520	020040	DH115:	.ASCIZ / PC		HITMIS/				
13859	113464	020040	044510	046524								
13860	113472	051511	000									
13861												
13862	113475	040	050040	020103	DH123:	.ASCIZ ? PC		KIPDR	(KIPDR)?			
13863	113502	020040	045440	050111								
13864	113510	051104	020040	045450								
13865	113516	050111	051104	000051								
13866	113524	020040	041520	020040	DH125:	.ASCIZ / PC	CCR	PAR-ADR	(PAR)	(PDR)	TST-DATA-ADRS(VA)/	
13867	113532	020040	041503	020122								
13868	113540	050040	051101	040455								
13869	113546	051104	020040	024040								
13870	113554	040520	024522	020040								
13871	113562	050050	051104	020051								
13872	113570	052040	052123	042055								
13873	113576	052101	026501	042101								
13874	113604	051522	053050	024501								
13875	113612	000										
13876	113613	040	050040	020103	DH130:	.ASCIZ ? PC		SIPDR	(SIPDR)?			
13877	113620	020040	051440	050111								
13878	113626	051104	020040	051450								
13879	113634	050111	051104	000051								
13880	113642	020040	041520	020040	DH132:	.ASCIZ ? PC		UIPDR	(UIPDR)?			
13881	113650	020040	044525	042120								
13882	113656	020122	024040	044525								
13883	113664	042120	024522	000								
13884												
13885												
13886	113671	040	052040	051505	DH136:	.ASCIZ ' TEST.'	<TAB>	'CALL AT PC.'	<TAB>	'GROUP.'	<TAB>	'ADDRESS.'
13887	113676	027124	041411	046101								
13888	113704	020114	052101	050040								
13889	113712	027103	020011	051107								
13890	113720	052517	027120	040411								
13891	113726	042104	042522	051523								
13892	113734	000056										
13893												
13894		113671										
13895												
13896	113736	020040	042524	052123	DH140:	.ASCIZ ' TEST.'	<TAB>	'CALL AT PC.'	<TAB>	'DATA.'	<TAB>	'ADDRESS.'
13897	113744	004456	040503	046114								
13898	113752	040440	020124	041520								
13899	113760	004456	040504	040524								
13900	113766	004456	042101	051104								

DH137=DH136

13901	113774	051505	027123	000	
13902					
13903		113736			DH141=DH140
13904					
13905		113736			DH142=DH140
13906					
13907		113736			DH143=DH140
13908					
13909		113736			DH144=DH140
13910					
13911		113736			DH145=DH140
13912					
13913		113736			DH146=DH140
13914					
13915		113736			DH147=DH140
13916					
13917	114001	040	052040	051505	DH150: .ASCIZ ' TEST.<TAB>'TRAP AT PC.<TAB>'CALL AT PC.<TAB>'CPU ERROR REGISTER.'
13918	114006	027124	052011	040522	
13919	114014	020120	052101	050040	
13920	114022	027103	041411	046101	
13921	114030	020114	052101	050040	
13922	114036	027103	041411	052520	
13923	114044	042440	051122	051117	
13924	114052	051040	043505	051511	
13925	114060	042524	027122	000	
13926					
13927	114065	125	044523	043516	DH151: .ASCII 'USING THE RS04.'
13928	114072	052040	042510	051040	
13929	114100	030123	027064		
13930	114104	020040	042524	052123	.ASCIZ ' TEST.<TAB>'GROUP.<TAB>'ADDRESS.'
13931	114112	004456	051107	052517	
13932	114120	027120	040411	042104	
13933	114126	042522	051523	000056	
13934					
13935	114134	051525	047111	020107	DH152: .ASCII 'USING THE RP04.'
13936	114142	044124	020105	050122	
13937	114150	032060	056		
13938	114153	040	052040	051505	.ASCIZ ' TEST.<TAB>'GROUP.<TAB>'ADDRESS.'
13939	114160	027124	043411	047522	
13940	114166	050125	004456	042101	
13941	114174	051104	051505	027123	
13942	114202	000			
13943					
13944	114203	125	044523	043516	DH153: .ASCII 'USING THE MASS BUS TESTER.'
13945	114210	052040	042510	046440	
13946	114216	051501	020123	052502	
13947	114224	020123	042524	052123	
13948	114232	051105	056		
13949	114235	040	052040	051505	.ASCIZ ' TEST.<TAB>'GROUP.<TAB>'ADDRESS.'
13950	114242	027124	043411	047522	
13951	114250	050125	004456	042101	
13952	114256	051104	051505	027123	
13953	114264	000			
13954					
13955	114265	040	052040	051505	DH154: .ASCIZ ' TEST.<TAB>'RS4CS2.<TAB>'RS4DS.<TAB>'RS4ER.'
13956	114272	027124	051011	032123	

13957	114300	051503	027062	051011	
13958	114306	032123	051504	004456	
13959	114314	051522	042464	027122	
13960	114322	000			
13961					
13962	114323	040	052040	051505	DH155: .ASCIZ ' TEST.'<TAB>'RP4CS2.'<TAB>'RP4DS.'<TAB>'RP4ER.'
13963	114330	027124	051011	032120	
13964	114336	051503	027062	051011	
13965	114344	032120	051504	004456	
13966	114352	050122	042464	027122	
13967	114360	000			
13968					
13969	114361	040	052040	051505	DH156: .ASCIZ ' TEST.'<TAB>'RH4CS2.'<TAB>'RH4ST.'<TAB>'RH4ER.'
13970	114366	027124	051011	032110	
13971	114374	051503	027062	051011	
13972	114402	032110	052123	004456	
13973	114410	044122	042464	027122	
13974	114416	000			
13975					
13976					
13977	114417	040	052040	051505	DH160: .ASCIZ ' TEST.'<TAB>'RK5ER.'<TAB>'RK5DS.'
13978	114424	027124	051011	032513	
13979	114432	051105	004456	045522	
13980	114440	042065	027123	000	
13981					
13982	114445	040	052040	051505	DH161: .ASCIZ ' TEST.'<TAB>'UBECR1.'<TAB>'UBECR2.'
13983	114452	027124	052411	042502	
13984	114460	051103	027061	052411	
13985	114466	042502	051103	027062	
13986	114474	000			
13987	114475	105	051122	051117	DH162: .ASCIZ /ERRORPC TEST NUMBER/
13988	114502	041520	052040	051505	
13989	114510	020124	052516	041115	
13990	114516	051105	000		

:THESE ARE DATA FORMAT DESIGNATORS FOR THE DATA TABLE:

13991					
13992					
13993					
13994	114521	004	004	003	DF1: .BYTE 4,4,3,3
13995	114524	003			
13996					
13997	114525	004	004	007	DF2: .BYTE 4,4,7,0,3,3
13998	114530	000	003	003	
13999					
14000	114525				DF3=DF2
14001					
14002	114525				DF4=DF2
14003					
14004	114533	004	003	000	DF5: .BYTE 4,3,0,5,0,0,0,0
14005	114536	005	000	000	
14006	114541	000	000		
14007					
14008	114533				DF6=DF5
14009					
14010	114533				DF7=DF5
14011					
14012	114533				DF10=DF5



14013							
14014	114543	004	004	003	DF11:	.BYTE	4,4,3,7,5,0,5,3,0
14015	114546	007	005	000			
14016	114551	005	003	000			
14017							
14018	114554	004	004	003	DF12:	.BYTE	4,4,3,3,0,0
14019	114557	003	000	000			
14020							
14021	114562	004	004	000	DF13:	.BYTE	4,4,0,0,4
14022	114565	000	004				
14023							
14024	114567	004	003	007	DF14:	.BYTE	4,3,7,3,0
14025	114572	003	000				
14026							
14027	114574	004	003		DF15:	.BYTE	4,3
14028							
14029	114576	004	004	000	DF16:	.BYTE	4,4,0,0,3,3
14030	114601	000	003	003			
14031							
14032	114604	004	004	003	DF17:	.BYTE	4,4,3,0,5,3,5,5,5,3,5,3,5,3,5,3,5,0,5,0,5,0,5,0
14033	114607	000	005	003			
14034	114612	005	005	005			
14035	114615	003	005	003			
14036	114620	005	003	005			
14037	114623	003	005	000			
14038	114626	005	000	005			
14039	114631	000	005	000			
14040							
14041	114604				DF20=DF17		
14042							
14043	114634	004	003	000	DF21:	.BYTE	4,3,0,4,3,5
14044	114637	004	003	005			
14045	114642	005	003	005		.BYTE	5,3,5,3,5,3,5,3,5
14046	114645	003	005	003			
14047	114650	005	003	005			
14048	114653	000	005	000		.BYTE	0,5,0,5,0,5,0
14049	114656	005	000	005			
14050	114661	000					
14051							
14052	114662	004	003	002	DF22:	.BYTE	4,3,2
14053							
14054	114665	004	003	002	DF23:	.BYTE	4,3,2,2,0
14055	114670	002	000				
14056							
14057	114662				DF24=DF22		
14058							
14059	114665				DF25=DF23		
14060							
14061	114672	004	003	004	DF26:	.BYTE	4,3,4,2
14062	114675	002					
14063							
14064	114676	004	003	004	DF27:	.BYTE	4,3,4,2,2
14065	114701	002	002				
14066							
14067	114662				DF30=DF22		
14068							

14069		114665				DF31=DF23
14070						
14071		114662				DF32=DF22
14072						
14073		114665				DF33=DF23
14074						
14075	114703	004	003	000		DF34: .BYTE 4,3,0,2,0
14076	114706	002	000			
14077						
14078		114703				DF35=DF34
14079						
14080	114710	004	003	004		DF36: .BYTE 4,3,4,2
14081	114713	002				
14082						
14083	114714	004	004	007		DF37: .BYTE 4,4,7,2,2,0
14084	114717	002	002	000		
14085						
14086						
14087	114722	004	003	004		DF41: .BYTE 4,3,4,2
14088	114725	002				
14089						
14090		114722				DF42=DF41
14091						
14092	114726	004	003	003		DF43: .BYTE 4,3,3,4,5,2,7,0
14093	114731	004	005	002		
14094	114734	007	000			
14095						
14096		114726				DF40=DF43
14097						
14098	114736	004	003	002		DF44: .BYTE 4,3,2,7,0,5,2,5,0,5,2,5,0,5,2
14099	114741	007	000	005		
14100	114744	002	005	000		
14101	114747	005	002	005		
14102	114752	000	005	002		
14103						
14104		114736				DF45=DF44
14105						
14106	114755	004	003	005		DF46: .BYTE 4,3,5,2,5,0,5,2,5,0,5,2
14107	114760	002	005	000		
14108	114763	005	002	005		
14109	114766	000	005	002		
14110						
14111		114755				DF47=DF46
14112						
14113		114736				DF50=DF44
14114						
14115		114736				DF51=DF44
14116						
14117		114755				DF52=DF46
14118						
14119		114755				DF53=DF46
14120						
14121	114771	004	003	004		DF54: .BYTE 4,3,4
14122	114774	000	000			DF55: .BYTE 0,0
14123		114774				DF56=DF55
14124		114774				DF57=DF55

14125		114774				DF60=DF55	
14126		114774				DF61=DF55	
14127		114774				DF62=DF55	
14128		114774				DF63=DF55	
14129		114774				DF64=DF55	
14130		114774				DF65=DF55	
14131	114776	000	000	000		DF66: .BYTE	0,0,0,0
14132	115001	000					
14133		114776				DF67=DF66	
14134		114776				DF70=DF66	
14135		114776				DF71=DF66	
14136		114776				DF72=DF66	
14137		114776				DF73=DF66	
14138		114776				DF74=DF66	
14139		114776				DF75=DF66	
14140		114776				DF76=DF66	
14141		114776				DF77=DF66	
14142							
14143	115002	000	000	000		DF100: .BYTE	0,0,0,0,0,0
14144	115005	000	000	000			
14145		114774				DF103=DF55	
14146		114774				DF104=DF55	
14147		114774				DF105=DF55	
14148		114774				DF106=DF55	
14149	115010	000	000	000		DF107: .BYTE	0,0,0,0,0
14150	115013	000	000				
14151							
14152		114774				DF110=DF55	
14153		114774				DF111=DF55	
14154		114774				DF112=DF55	
14155		114776				DF113=DF66	
14156		114774				DF115=DF55	
14157	115015	000	000	000		DF123: .BYTE	0,0,0
14158	115020	004	003	004		DF136: .BYTE	4,3,4,2
14159	115023	002					
14160		115020				DF137=DF136	
14161							
14162	115024	004	003	000		DF140: .BYTE	4,3,0,2
14163	115027	002					
14164							
14165		115024				DF141=DF140	
14166							
14167		115024				DF142=DF140	
14168							
14169		115024				DF143=DF140	
14170							
14171		115024				DF144=DF140	
14172							
14173		115024				DF145=DF140	
14174							
14175		115024				DF146=DF140	
14176							
14177		115024				DF147=DF140	
14178							
14179	115030	004	003	003		DF150: .BYTE	4,3,3,0
14180	115033	000					

```

14181
14182 115034 004 004 007 DF151: .BYTE 4,4,7
14183
14184 115034 DF152=DF151
14185 115034 DF153=DF151
14186
14187 115037 004 000 000 DF154: .BYTE 4,0,0,0
14188 115042 000
14189
14190 115037 DF155=DF154
14191 115037 DF156=DF154
14192 115037 DF157=DF154
14193 115037 DF160=DF154
14194 115037 DF161=DF154
14195
14196
14197 115044 .EVEN
14198
14199 ;THESE ARE DATA TABLES:
14200
14201 115044 001632 001634 001636 DT1: .WORD $TMP0,$TMP1,$TMP2,$ERRPC,0
14202 115052 001516 000000
14203
14204 115056 001632 001646 001636 DT2: .WORD $TMP0,$TMP6,$TMP2,$TMP1,$TMP5,$TMP4,0
14205 115064 001634 001644 001642
14206 115072 000000
14207
14208 115056 DT3=DT2
14209
14210 115056 DT4=DT2
14211
14212 115074 001632 001516 001636 DT5: .WORD $TMP0,$ERRPC,$TMP2,MTA5,JJPAT1,JJPAT2,JJPAT3,JJPAT4,0
14213 115102 066502 026276 026300
14214 115110 026302 026304 000000
14215
14216 115074 DT6=DT5
14217
14218 115074 DT7=DT5
14219
14220 115074 DT10=DT5
14221
14222 115116 001632 001634 001636 DT11: .WORD $TMP0,$TMP1,$TMP2,$TMP4,MTA11,$TMP3,$TAB,$TMP7,$TMP6,0
14223 115124 001642 066564 001640
14224 115132 066500 001650 001646
14225 115140 000000
14226
14227 115142 001632 001634 001516 DT12: .WORD $TMP0,$TMP1,$ERRPC,$TMP3,$TMP4,$TMP5,0
14228 115150 001640 001642 001644
14229 115156 000000
14230
14231 115160 001632 001634 001636 DT13: .WORD $TMP0,$TMP1,$TMP2,$TMP3,$TMP4,0
14232 115166 001640 001642 000000
14233
14234 115174 001632 001516 001634 DT14: .WORD $TMP0,$ERRPC,$TMP1,$TMP3,$TMP4,0
14235 115202 001640 001642 000000
14236
    
```

14237	115210	001632	001634	000000	DT15:	.WORD	\$TMP0,\$TMP1,0
14238							
14239	115216	001632	001634	001636	DT16:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,\$TMP4,\$ERRPC,0
14240	115224	001640	001642	001516			
14241	115232	000000					
14242							
14243	115234	001632	001634	001636	DT17:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,MTA17,\$TMP4,\$CRLF,MTB17
14244	115242	001640	066656	001642			
14245	115250	001713	066636				
14246	115254	066631	001644	066631	.WORD		MTA17,\$TMP5,MTA17,\$TMP6,MTA17,\$TMP7,MTA17,\$TMP10
14247	115262	001646	066631	001650			
14248	115270	066631	001652				
14249	115274	001713	035542	066500	.WORD		\$CRLF,MMPAT1,\$TAB,MMPAT2,\$TAB,MMPAT3,\$TAB,MMPAT4,0
14250	115302	035544	066500	035546			
14251	115310	066500	035550	000000			
14252							
14253	115316	001632	001634	001636	DT20:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,MTA20,\$TMP4,\$CRLF,MTB17
14254	115324	001640	066665	001642			
14255	115332	001713	066636				
14256	115336	001644	066631	001646	.WORD		\$TMP5,MTA17,\$TMP6,MTA17,\$TMP7,MTA17,\$TMP10,MTA17
14257	115344	066631	001650	066631			
14258	115352	001652	066631				
14259	115356	001713	035542	066500	.WORD		\$CRLF,MMPAT1,\$TAB,MMPAT3,\$TAB,MMPAT3,\$TAB,MMPAT4,0
14260	115364	035546	066500	035546			
14261	115372	066500	035550	000000			
14262							
14263	115400	001632	001634	001636	DT21:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,\$TMP4,MTA21
14264	115406	001640	001642	066674			
14265	115414	066631	001644	066631	.WORD		MTB21,\$TMP5,MTB21,\$TMP6,MTB21,\$TMP7,MTB21,\$TMP10,\$CRLF
14266	115422	001646	066631	001650			
14267	115430	066631	001652	001713			
14268	115436	033676	066500	033700	.WORD		KKPAT1,\$TAB,KKPAT2,\$TAB,KKPAT3,\$TAB,KKPAT4,0
14269	115444	066500	033702	066500			
14270	115452	033704	000000				
14271							
14272	115456	001632	001516	007632	DT22:	.WORD	\$TMP0,\$ERRPC,XADR2,0
14273	115464	000000					
14274							
14275	115466	001632	001516	007632	DT23:	.WORD	\$TMP0,\$ERRPC,XADR2,\$TMP3,\$TMP1,0
14276	115474	001640	001634	000000			
14277							
14278	115502	001632	001516	010532	DT24:	.WORD	\$TMP0,\$ERRPC,XXADR2,0
14279	115510	000000					
14280							
14281	115512	001632	001516	010532	DT25:	.WORD	\$TMP0,\$ERRPC,XXADR2,\$TMP3,\$TMP1,0
14282	115520	001640	001634	000000			
14283							
14284	115526	001632	001516	001634	DT26:	.WORD	\$TMP0,\$ERRPC,\$TMP1,\$TMP2,0
14285	115534	001636	000000				
14286							
14287	115540	001632	001516	001634	DT27:	.WORD	\$TMP0,\$ERRPC,\$TMP1,\$TMP2,\$TMP4,0
14288	115546	001636	001642	000000			
14289							
14290	115554	001632	001516	011450	DT30:	.WORD	\$TMP0,\$ERRPC,RRADR2,0
14291	115562	000000					
14292							

14293	115564	001632	001516	011450	DT31:	.WORD	\$TMP0,\$ERRPC,\$RRADR2,\$TMP3,\$TMP1,0
14294	115572	001640	001634	000000			
14295							
14296	115600	001632	001516	012332	DT32:	.WORD	\$TMP0,\$ERRPC,\$SADR2,0
14297	115606	000000					
14298							
14299	115610	001632	001516	012332	DT33:	.WORD	\$TMP0,\$ERRPC,\$SADR2,\$TMP3,\$TMP1,0
14300	115616	001640	001634	000000			
14301							
14302	115624	001632	001516	001636	DT34:	.WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,\$TMP5,0
14303	115632	001640	001644	000000			
14304							
14305		115624			DT35=DT34		
14306							
14307	115640	001632	001516	001636	DT36:	.WORD	\$TMP0,\$ERRPC,\$TMP2,\$BBADR1,0
14308	115646	015136	000000				
14309							
14310	115652	001632	001634	015156	DT37:	.WORD	\$TMP0,\$TMP1,\$BBCNT1,\$BBADR2,\$BBADR3,0
14311	115660	015142	015146	000000			
14312							
14313							
14314	115666	001632	001516	001636	DT41:	.WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,0
14315	115674	001640	000000				
14316							
14317		115666			DT42=DT41		
14318							
14319	115700	001632	001516	001636	DT43:	.WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,\$MTA43,\$TMP5,\$TMP7,\$TMP4,0
14320	115706	001640	066761	001644			
14321	115714	001650	001642	000000			
14322							
14323		115700			DT40=DT43		
14324							
14325	115722	001632	001516	001666	DT44:	.WORD	\$TMP0,\$ERRPC,\$TMP16,\$TMP3,\$TMP5,\$MTA45,\$TMP12,\$MTB45
14326	115730	001640	001644	067034			
14327	115736	001656	067062				
14328	115742	001652	067077	001646		.WORD	\$TMP10,\$MTC45,\$TMP6,\$MTB45,\$TMP11,\$MTC45,\$TMP14,0
14329	115750	067062	001654	067077			
14330	115756	001662	000000				
14331							
14332		115722			DT45=DT44		
14333							
14334	115762	001632	001656	067034	DT46:	.WORD	\$TMP0,\$TMP12,\$MTA45,\$TMP10,\$MTB45,\$TMP6,\$MTC45
14335	115770	001652	067062	001646			
14336	115776	067077					
14337	116000	001636	067062	001650		.WORD	\$TMP2,\$MTB45,\$TMP7,\$MTC45,\$TMP4,0
14338	116006	067077	001642	000000			
14339							
14340		115762			DT47=DT46		
14341							
14342	116014	001632	001516	001666	DT50:	.WORD	\$TMP0,\$ERRPC,\$TMP16,\$TMP3,\$TMP5,\$MTA50,\$TMP12,\$MTB45
14343	116022	001640	001644	067112			
14344	116030	001656	067062				
14345	116034	001652	067077	001646		.WORD	\$TMP10,\$MTC45,\$TMP6,\$MTB45,\$TMP11,\$MTC45,\$TMP14,0
14346	116042	067062	001654	067077			
14347	116050	001662	000000				
14348							

14349		116014			DT51=DT50	
14350						
14351	116054	001632	001656	067112	DT52: .WORD	\$TMP0,\$TMP12,MTA50,\$TMP10,MTB45,\$TMP6,MTC45
14352	116062	001652	067062	001646		
14353	116070	067077				
14354	116072	001636	067062	001650	.WORD	\$TMP2,MTB45,\$TMP7,MTC45,\$TMP4,0
14355	116100	067077	001642	000000		
14356						
14357		116054			DT53=DT52	
14358						
14359	116106	001632	001516	001636	DT54: .WORD	\$TMP0,\$ERRPC,\$TMP2,0
14360	116114	000000				
14361	116116	001516	001562	000000	DT55: .WORD	\$ERRPC,\$REG0,0
14362	116124	001516	001562	001564	DT66: .WORD	\$ERRPC,\$REG0,\$REG1,\$REG2,0
14363	116132	001566	000000			
14364	116136	001516	001562	001564	DT100: .WORD	\$ERRPC,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4,0
14365	116144	001566	001570	001572		
14366	116152	000000				
14367	116154	001516	001562	001564	DT107: .WORD	\$ERRPC,\$REG0,\$REG1,\$REG2,\$REG3,0
14368	116162	001566	001570	000000		
14369						
14370	116170	001516	001562	001564	DT123: .WORD	\$ERRPC,\$REG0,\$REG1,0
14371	116176	000000				
14372	116200	001516	001562	001564	DT125: .WORD	\$ERRPC,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4,0
14373	116206	001566	001570	001572		
14374	116214	000000				
14375						
14376	116216	001632	001516	001636	DT136: .WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,0
14377	116224	001640	000000			
14378						
14379		116216			DT137=DT136	
14380						
14381	116230	001632	001516	001636	DT140: .WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,0
14382	116236	001640	000000			
14383						
14384		116230			DT141=DT140	
14385						
14386		116230			DT142=DT140	
14387						
14388		116230			DT143=DT140	
14389						
14390		116230			DT144=DT140	
14391						
14392		116230			DT145=DT140	
14393						
14394		116230			DT146=DT140	
14395						
14396		116230			DT147=DT140	
14397						
14398	116242	001632	001634	001636	DT150: .WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,0
14399	116250	001640	000000			
14400						
14401	116254	001632	001634	001636	DT151: .WORD	\$TMP0,\$TMP1,\$TMP2,0
14402	116262	000000				
14403						
14404		116254			DT152=DT151	

```
14405      116254      DT153=DT151
14406  116264  001632  001634  001636  DT154:  .WORD  $TMP0,$TMP1,$TMP2,$TMP3,0
14407  116272  001640  000000
14408      116264      DT155=DT154
14409      116264      DT156=DT154
14410      116254      DT157=DT151
14411      116254      DT160=DT151
14412      116254      DT161=DT151
14413  116276  001516  001502  000000  DT162:  .WORD  $ERRPC,$STNM,0
14414
14415      116304      TLOC=.
14416      116304      TLOC=-4&TLOC
14417      116310      TLOC=TLOC+4
14418      116310      .=TLOC
14419  116310  001000  TSTDAT: .BLKW  512.
14420
14421
14422
14423  120310  000000  000000  000000  BOTTOM:  .WORD  0,0,0
14424      126310      BOTPRG=BOTTOM+6000
14425      000001      .END
```











DDPER	025152	4877	4908#						
DDPER1	025176	4909	4914#						
DDPU1	025112	4896#							
DDPU2	025136	4898	4903#						
DDPV	025066	4888	4890#						
DDTMP	025202	4879*	4891	4896		4916#			
DD1	025020	4874	4877#						
DF1	114521	648	13994#						
DF10 =	114533	669	14012#						
DF100 =	115002	1053	1060	1067		14143#			
DF103 =	114774	943	14145#						
DF104 =	114774	950	14146#						
DF105 =	114774	957	14147#						
DF106 =	114774	964	14148#						
DF107	115010	971	14149#						
DF11	114543	672	14014#						
DF110 =	114774	978	14152#						
DF111 =	114774	985	14153#						
DF112 =	114774	992	14154#						
DF113 =	114776	999	14155#						
DF115 =	114774	1011	14156#						
DF12	114554	675	14018#						
DF123	115015	1042	1047	1072		1078	1085	1091	14157#
DF13	114562	678	14021#						
DF136	115020	1099	14158#	14160					
DF137 =	115020	1102	14160#						
DF14	114567	681	14024#						
DF140	115024	1105	14162#	14165		14167	14169	14171	14173 14175 14177
DF141 =	115024	1108	14165#						
DF142 =	115024	1111	14167#						
DF143 =	115024	1114	14169#						
DF144 =	115024	1117	14171#						
DF145 =	115024	1120	14173#						
DF146 =	115024	1123	14175#						
DF147 =	115024	1126	14177#						
DF15	114574	684	14027#						
DF150	115030	1129	14179#						
DF151	115034	1132	14182#	14184		14185			
DF152 =	115034	1135	14184#						
DF153 =	115034	1138	14185#						
DF154	115037	1141	14187#	14190		14191	14192	14193	14194
DF155 =	115037	1144	14190#						
DF156 =	115037	1147	14191#						
DF157 =	115037	14192#							
DF16	114576	687	14029#						
DF160 =	115037	1153	14193#						
DF161 =	115037	1156	14194#						
DF17	114604	690	14032#	14041					
DF2	114525	651	13997#	14000		14002			
DF20 =	114604	693	14041#						
DF21	114634	696	14043#						
DF22	114662	699	14052#	14057		14067	14071		
DF23	114665	702	14054#	14059		14069	14073		
DF24 =	114662	705	14057#						
DF25 =	114665	708	14059#						
DF26	114672	711	14061#						











EE10	023510	4601	4605#	4660	
EE11	023540	4612#	4615		
EE12	023546	4611	4614#		
EE13	023552	4613	4617#		
EE2	023370	4576#	4579		
EE3	023376	4575	4578#		
EE4	023404	4577	4581#	4652	
EE5	023432	4588#	4591		
EE6	023440	4587	4590#		
EE7	023446	4589	4593#	4656	
EE8	023476	4600#	4603		
EE9	023504	4599	4602#		
EMTVEC=	000030	147#	1302*	1303*	
EM1	072022	648	12083#		
EM10	072762	669	12174#		
EM100	101464	12837#			
EM103	101524	940	12843#		
EM104	101605	947	12852#		
EM105	101711	954	12864#		
EM106	101765	961	12872#		
EM107	102055	968	12882#		
EM11	073053	672	12186#		
EM110	102174	975	12896#		
EM111	102222	982	12900#		
EM112	102304	989	12909#		
EM113	102345	996	12915#		
EM115	102403	1008	12921#		
EM12	073141	675	12197#		
EM123	102520	1039	12936#		
EM124	102566	1044	12943#		
EM125	102630	1050	12949#		
EM126	102670	1056	12955#		
EM127	103021	1063	12970#		
EM13	073266	678	12214#		
EM130	103150	1069	12986#		
EM131	103216	1075	12993#		
EM132	103260	1082	12999#		
EM133	103326	1088	13006#		
EM136	103370	1099	13012#		
EM137	103605	1102	13037#		
EM14	073346	681	12224#		
EM140	104023	1105	13063#		
EM141	104364	1108	13105#		
EM142	104724	1111	13147#		
EM143	105266	1114	13189#		
EM144	105627	1117	13231#		
EM145	106161	1120	13271#		
EM146	106512	1123	13312#		
EM147	107045	1126	13353#		
EM15	073405	684	12231#		
EM150	107377	1129	13393#		
EM151	107463	1132	13403#	13413	13414
EM152 =	107463	1135	13413#		
EM153 =	107463	1138	13414#		
EM154	107544	1141	13416#		
EM155	107576	1144	13422#		

EM156	107630	1147	13428#			
EM16	073455	687	12240#			
EM160	107675	1153	13437#			
EM161	107727	1156	13443#			
EM162	107775	1160	13450#			
EM163	110205	1164	13475#			
EM164	110277	1168	13486#			
EM165	110356	1172	13494#			
EM166	110375	1176	13497#			
EM167	110461	1180	13506#			
EM17	073531	690	12249#	12259		
EM170	110565	1184	13518#			
EM171	110630	1188	13524#			
EM172	110674	1192	13530#			
EM173	110760	1196	13540#			
EM174	111146	1200	13561#			
EM175	111211	1204	13567#			
EM2	072107	651	12093#			
EM20 =	073531	693	12259#			
EM21 =	073612	696	12261#			
EM22	073676	699	12271#	12315	12358	12378
EM23	074112	702	12297#	12317		
EM24 =	073676	705	12315#			
EM25 =	074112	708	12317#			
EM26	074246	711	12319#			
EM27	074413	714	12338#			
EM3	072301	654	12117#			
EM30 =	073676	717	12358#			
EM31 =	074560	720	12360#	12380		
EM32 =	073676	723	12378#			
EM33 =	074560	726	12380#			
EM34	074717	729	12382#			
EM35	075023	732	12395#			
EM36	075132	735	12409#			
EM37	075264	738	12426#			
EM4	072415	657	12131#			
EM40	075346	741	12436#			
EM41	075521	744	12457#			
EM42	075705	747	12479#			
EM43	076160	750	12514#			
EM435	111026	13547#				
EM44	076306	753	12531#	12610		
EM45	076502	756	12554#	12612		
EM46	076701	759	12578#	12614		
EM47	077022	762	12594#	12616		
EM5	072530	660	12145#			
EM50 =	076306	765	12610#			
EM51 =	076502	768	12612#			
EM52 =	076701	771	12614#			
EM53 =	077022	774	12616#			
EM54	077146	777	12618#			
EM55	077321	782	12637#			
EM56	077352	791	12642#			
EM57	077420	799	12649#			
EM6	072610	663	12154#			
EM60	077461	807	12655#			

























RH4HH	040325	7257*	7311#							
RH4H1	063554	11568	11572#							
RH4H2	063670	11592#								
RH4H3	064000	11606	11610#							
RH4H4	064014	11600	11615#							
RH4H5	064030	11610	11616	11621#						
RH4H51	064032	11622#	11637	11639						
RH4H6	064116	11625	11627	11636#						
RH4II	040326	7264*	7312#							
RH4JJ	040330	7313#								
RH4KK	040332	7269*	7314#							
RH4LL	040334	7276*	7315#							
RH4MA1	063524	11559#	11581*	11594						
RH4MA2	063526	11560#	11582*	11595						
RH4MM	040356	7317	7324#							
RH4MR1	004064	1257#	11597*							
RH4MR2	004046	1250#								
RH4NN	040371	7258*	7331#							
RH4OO	040372	7265*	7332#							
RH4PP	040374	7333#								
RH4QQ	040376	7270*	7334#							
RH4RB	036276	6870#	7241	7242						
RH4RDY	064160	11589	11650#							
RH4REG	004036	1246#	10677							
RH4REX	004070	1260#	10684							
RH4RR	040400	7272*	7335#							
RH4SEC	063536	11564#								
RH4SS	040464	7355	7361#							
RH4ST	004052	1252#	11629	11638	11665	11680				
RH4SUN	036252	6857#	7254							
RH4S1	064134	11588	11643#							
RH4S2	064144	11590	11646#							
RH4TMP	063512	11554#	11573*	11576						
RH4TRK	063534	11563#								
RH4UNI	063516	11556#	11578*	11592	11622	11643*	11651	11653	11656	
RH4USE	063510	11553#	11650*	11651*	11652	11687				
RH4V	004140	1285#	11599							
RH4VEC	063532	11562#	11584*	11617						
RH4WC	004042	1248#	11593*							
RH4WCT	063530	11561#	11583*	11593	11647*					
RH4XX	040502	7365	7368#							
RH4YY	040510	7300	7306	7320	7325	7340	7344	7357	7362	7371#
RH4ZZ	040560	7373	7386#							
RH4111	040422	7337	7344#							
RH4112	040433	7255*	7349#							
RH4113	040434	7263*	7350#							
RH4114	040436	7351#								
RH4115	040440	7271*	7352#							
RH4116	040442	7273*	7353#							
RK5AA	040612	7402#	7538							
RK5AA1	041052	7436*	7451#							
RK5AA2	041054	7443*	7452#							
RK5AA3	041050	7400*	7450#	7454*						
RK5ASS	036266	6865#								
RK5BA	004110	1269#	11290*							
RK5BB	041073	7421*	7460#							

RK5CC	041074	7427*	7461#											
RK5CLR	062730	11325	11407	11411#										
RK5CR	036242	6852#	6878	7533*	7545*									
RK5CS1	004104	1267#	11287*	11291*	11298*	11319	11380*	11381	11385*	11387	11412*	11413		
RK5CYL	062000	11252#	11350*											
RK5DA	004112	1270#	11288*	11292*	11379*	11384*	11411*							
RK5DA1	061760	11244#	11266*	11292	11342	11360*								
RK5DA2	061762	11245#	11267*											
RK5DB	004114	1271#												
RK5DD	041076	7431*	7462#											
RK5DFL	057463	10648*	10719#	10836*										
RK5DS	004100	1265#	10693	11323	11331	11392	11405							
RK5EE	041100	7437*	7463#											
RK5ER	004102	1266#	11322	11395	11404									
RK5ER1	061740	7541	10642*	10829	11236#	11273*	11324*	11406*						
RK5ER2	061742	7546	11237#	11274*	11322*	11404*								
RK5ER3	061744	7548	11238#	11275*	11323*	11405*								
RK5ER4	061746	7547	11239#											
RK5FF	041102	7444*	7464#											
RK5FLG	061736	10637*	11235#	11254	11297*	11307*	11312*							
RK5FT	043115	7916#												
RK5FUN	061754	11242#	11264*	11296										
RK5GG	041124	7466	7474#											
RK5HAN	062002	10037	11254#											
RK5HH	041137	7422*	7480#											
RK5H1	062014	11255	11259#											
RK5H2	062134	11287#												
RK5H3	062240	11302	11306#											
RK5H4	062254	11294	11311#											
RK5H5	062302	11306	11311	11318#										
RK5H51	062304	11319#	11330	11332										
RK5H6	062346	11321	11329#											
RK5II	041140	7429*	7481#											
RK5JJ	041142	7432*	7482#											
RK5KK	041144	7438*	7483#											
RK5LL	041146	7445*	7484#											
RK5MA1	061764	11246#	11268*	11290	11365	11376*								
RK5MA2	061766	11247#	11269*	11291	11366	11375*								
RK5MM	041170	7486	7493#											
RK5NN	041203	7423*	7500#											
RK5OO	041204	7430*	7501#											
RK5PP	041206	7434*	7502#											
RK5QQ	041210	7439*	7503#											
RK5RB	036300	6871#	7406	7407										
RK5RDY	062570	11278	11379#											
RK5REG	004076	1264#	10695											
RK5RR	041212	7441*	7504#											
RK5SEC	061776	11251#	11351*											
RK5SS	041276	7524	7530#											
RK5SUN	036254	6858#	7419											
RK5S1	062364	11277	11336#											
RK5S2	062406	11280	11342#											
RK5S3	062510	11283	11363#											
RK5TMP	061752	11241#	11260*	11263										
RK5TRK	061774	11250#	11352*											
RK5UNI	061756	11243#	11265*	11288	11336	11339*	11379	11384						





RP4H2	060230	10963#																		
RP4H3	060336	10976	10979#																	
RP4H4	060352	10970	10984#																	
RP4H5	060366	10979	10985	10989#																
RP4H51	060370	10990#	11004	11006																
RP4H6	060454	10993	10995	11003#																
RP4II	037534	7096*	7148#																	
RP4JJ	037536	7099*	7149#																	
RP4KK	037540	7105*	7150#																	
RP4LA	004002	1231#																		
RP4LL	037542	7112*	7151#																	
RP4MA1	060060	10928#	10949*	10965																
RP4MA2	060062	10929#	10950*	10966	11021*															
RP4MM	037564	7153	7160#																	
RP4MR	004006	1233#																		
RP4NN	037577	7090*	7167#																	
RP4OF	004014	1236#	11055*																	
RP4OO	037600	7097*	7168#																	
RP4PP	037602	7101*	7169#																	
RP4QQ	037604	7106*	7170#																	
RP4RB	036274	6869#	7073	7074																
RP4RDY	060542	10958	11024#																	
RP4REG	003760	1222#	10667																	
RP4RR	037606	7108*	7171#																	
RP4RR1	003776	1229#	10998	11062																
RP4RR2	004022	1239#																		
RP4RR3	004024	1240#																		
RP4SEC	060072	10933#																		
RP4SN	004012	1235#																		
RP4SS	037672	7191	7197#																	
RP4SUN	036250	6856#	7086																	
RP4S1	060472	10957	11010#																	
RP4S2	060502	10959	11013#																	
RP4S3	060526	10961	11020#																	
RP4TMP	060046	10923#	10941*	10944																
RP4TRK	060070	10932#																		
RP4UNI	060052	10925#	10946*	10963	10990	11010*	11025	11027	11030											
RP4USE	060044	10922#	11024*	11025*	11026	11068														
RP4V	004136	1284#	10969																	
RP4VEC	060066	10931#	10952*	10986																
RP4WC	003764	1224#	10964*																	
RP4WCT	060064	10930#	10951*	10964	11020*															
RP4XX	037710	7201	7204#																	
RP4YY	037716	7136	7142	7156	7161	7176	7180	7193	7198	7207#										
RP4ZZ	037766	7209	7222#																	
RP4111	037630	7173	7180#																	
RP4112	037641	7087*	7185#																	
RP4113	037642	7095*	7186#																	
RP4114	037644	7100*	7187#																	
RP4115	037646	7107*	7188#																	
RP4116	037650	7109*	7189#																	
RR =	000007	2294#																		
RRADR1	011444	2332*	2333*	2338	2340	2357	2358	2468*	2469*	2475#										
RRADR2	011450	2357*	2358*	2359*	2360*	2361*	2369	2371	2391	2392	2425	2427	2477#	14290						
		14293																		
RRADR3	011454	2330*	2331*	2338	2340	2359	2361	2457	2459	2461*	2464*	2465*	2470*	2471*						

































		5172*	5179*	5194*	5203*	5266*	5267*	5290*	5309*	5354*	5434*	5463*	5480*	5509*
		5588*	5617*	5634*	5663*	5757*	5758*	5919*	6021*	6035*	6036*	6050*	6064*	6065*
		6124*	6143*	6144*	6163*	6182*	6183*	6278*	6349*	6363*	6364*	6378*	6392*	6393*
		6424*	6441*	6460*	6477*	6496*	6513*	6532*	6549*	6575*	6576*	6598*	6599*	6621*
		6622*	6644*	6645*	6675*	6679*	6690*	6694*	7047*	7215*	7379*	7548*	7648*	7810*
		7850*	7874*	7897*	10226*	10243*	14201*	14204*	14212*	14222*	14231*	14239*	14243*	14253*
		14263	14284	14287	14302	14307	14314	14319	14337	14354	14359	14376	14381	14398
		14401	14406											
		584#	4337*	4472*	4633*	4793*								
\$TMP20	001672													
\$TMP21	001674													
\$TMP22	001676													
\$TMP23	001700													
\$TMP3	001640													
		571#	2015*	2249*	2449*	2648*	2722*	2738*	2830*	2855*	3051*	3071*	3338*	3529*
		3555*	3573*	3792*	3864*	3907*	4092*	4165*	4206*	4338*	4371*	4473*	4506*	4634*
		4667*	4794*	4827*	5180*	5204*	5310*	5355*	5381*	5464*	5510*	5618*	5664*	5920*
		6261*	6266*	6270*	6275*	6676*	6691*	7046*	7214*	7378*	7547*	7811*	7852*	7875*
		7898*	10227*	10244*	14222	14227	14231	14234	14239	14243	14253	14263	14275	14281
		14293	14299	14302	14314	14319	14325	14342	14376	14381	14398	14406		
\$TMP4	001642													
		572#	2016*	2250*	2450*	2649*	2723*	2739*	2831*	2856*	3072*	3341*	3556*	3574*
		3793*	3865*	3908*	3915*	3920	4093*	4166*	4207*	4214*	4219	4339*	4372*	4373*
		4474*	4507*	4508*	4635*	4668*	4669*	4795*	4828*	4829*	5181*	5205*	5311*	5357*
		5382*	5465*	5511*	5619*	5665*	5921*	6260*	6264*	6265*	6269*	6273*	6274*	6669*
		6673*	6682*	6683*	6687*	6688*	10241*	14204	14222	14227	14231	14234	14239	14243
		14253	14263	14287	14319	14337	14354							
\$TMP5	001644													
		573#	2719*	2736*	2827*	2853*	3073*	3344*	3909*	4208*	4340*	4374*	4475*	4509*
		4636*	4670*	4796*	4830*	5358*	5359*	5383*	5923*	5991*	6324*	14204	14227	14246
		14256	14265	14302	14319	14325	14342							
\$TMP6	001646													
		574#	3345*	3910*	4209*	4334*	4341*	4375*	4469*	4476*	4510*	4630*	4637*	4671*
		4790*	4797*	4831*	5360*	5924*	5992*	5993*	6325*	6326*	14204	14222	14246	14256
		14265	14328	14334	14345	14351								
\$TMP7	001650													
		575#	3342*	3911*	4210*	4342*	4376*	4477*	4511*	4638*	4672*	4798*	4832*	5356*
		5994*	6327*	14222	14246	14256	14265	14319	14337	14354				
\$TN	= 000061													
		10#	1512	1519#	1560	1570#	1603	1611#	1714	1725#	1784	1843#	1844	1845
		2051	2074#	2075	2076	2279	2293#	2294	2295	2484	2498#	2499	2500	2679
		2689#	2690	2691	2752	2761#	2762	2763	2870	2893#	2894	3162	3173#	3174
		3175	3375	3413#	3414	3415	3656	3681#	3682	3683	3952	3977#	3978	3979
		4249	4262#	4263	4264	4386	4395#	4396	4397	4520	4533#	4534	4535	4681
		4690#	4691	4692	4842	4866#	4867	4923	4931#	4932	5107	5115#	5116	5224
		5232#	5233	5391	5409#	5410	5411	5545	5563#	5564	5565	5700	5709#	5939
		5962#	5963	6289	6304#	6305	6701	6760#	7772	7782#	8016	8024#	8061	8069#
		8086	8091	8099#	8116	8125#	8146	8154#	8181	8201#	8295	8299	8315#	8374
		8382	8404#	8498	8502	8515#	8627	8635	8645#	8669	8690#	8776	8784	8799#
		8826	8835	8841	8848#	8953	8959	8974#	9037	9042#	9046	9073	9078#	9082
		9108	9113#	9117	9143	9148#								
\$TPB	001552													
\$TPFLG	001557													
\$TPS	001550													
\$TRAP	054374													
\$TRAP2	054416													
\$TRP	= 000033													
\$TRPAD	054430													
\$STSM	001404													
\$STSTM	001502													
		9997#	10008											
		10001#	10010#	10011#	10012#	10013#	10014#	10015	10016#	10017	10018#	10019#	10020#	10021#
		10022#	10023	10024#	10025#	10026#	10027#	10028#	10029#	10030#	10031#	10032#	10033#	10034
		10035#	10036#	10037#	10038#	10039#								
		9991	10008#											
		501#												
		518#	1291*	1848	2079	2298	2503	2694	2766	2897	3179	3418	3688	3984
		4267	4400	4538	4695	4870	4936	5120	5237	5414	5568	5714	5967	6309









.STYPO 1# 9795  
.S4OCA 1#  
.1170 1# 29

. ABS. 120316 000

ERRORS DETECTED: 0

CEKBDE.BIN,CEKBDE.LST/CRF/SOL=CEKBDE.SML,CEKBDE.P11  
RUN-TIME: 100 130 19 SECONDS  
RUN-TIME RATIO: 621/250=2.4  
CORE USED: 45K (89 PAGES)