

11/70-74

11/70 - 74 CACHE # 2
CEKBDD0

AH-7972D-MC

COPYRIGHT 75-80

FICHE 1 OF 2

JAN 1980

digital

MADE IN USA

This image shows a microfiche card with a grid of 14 columns and 16 rows of frames. Each frame contains a small, illegible image of a document page. The text in the frames is too small to be read, but the layout suggests a structured document, possibly a table or a list of entries. The card is labeled '11/70-74' and 'CEKBDD0' in the top left, and 'AH-7972D-MC', 'COPYRIGHT 75-80', and 'FICHE 1 OF 2' in the top right. The 'digital' logo and 'MADE IN USA' are also visible in the top right corner.

11/70-74

11/70 - 74 CACHE # 2
CEKBDD0

AH-7972D-MC

COPYRIGHT 75-80
FICHE 2 OF 2

JAN 1980

digital

MADE IN USA

This microfiche card contains a grid of frames. The left side of the card is filled with a grid of frames, each containing a small table of data. The data is organized into columns and rows, with some frames appearing to have headers. The right side of the card is mostly blank, with a few faint markings and a small vertical barcode-like structure near the bottom right corner.

IDENTIFICATION

SEQ 0001

PRODUCT CODE: AC-7971D-MC
PRODUCT NAME: CEKBDDO PDP-11/70-74MP CACHE DIAGNOSTIC (PART 2)
DATE: MAY, 1979
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 DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C): 1975, 1979 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	DECX/11

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

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 11/70-74MP 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.

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 TO 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 SETTINGS 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 AET (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 UBCB.

TEST 3 MEM MGT AND PE TRAP PRIORITY ARBITRATION

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

K T

SEQ 0010

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 CONTAINS 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 17777776

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 LINE 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 BIT 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 LIGIC. 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 LIGIC. 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 INVALIDATER LOGIC WORKED NO PARITY ERRORS CAN OCCUR. BUT IF THIS INVALIDATER FAILED THERE IS AN EXTREMELY HIGH PROBABILITY FOR THE OCCURENCE OF A CACHE DATA OR CACHE ADDRESS PARITY ERROR. IN FACT IF THE INVALIDATER 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.

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

MAKE 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' (VISU) 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 & VALID STORE SWITCHING

THIS TEST CHECKS THAT WHEN A CACHE FLUSH IS DONE BY SETTING CF IN CCR, THE VALID STORE IN USE (VSIU) SWITCHES. 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 & 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.

TEST 45 CHECK VALID STORES (A&B) FOR GROUPS 0 & 1

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 & 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

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

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 3YP 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

20	OPERATIONAL SWITCH SETTINGS
35	BASIC DEFINITIONS
160	CACHE REGISTER DEFINITIONS
171	CPU REGISTER DEFINITIONS
185	MEMORY MANAGEMENT DEFINITIONS
334	UNIBUS MAP REGISTER DEFINITIONS
472	TRAP CATCHER
479	STARTING ADDRESS(ES)
486	ACT11 HOOKS
513	COMMON TAGS
601	ERROR POINTER TABLE
1437	T1 PARITY ERROR ABORT
1486	T2 PARITY ERROR TRAP
1529	T3 MEM MGT AND PE TRAP PRIORITY ARBITRATION
1640	T4 UNIBUS PARITY ERROR
1711	T5 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ONES
1976	T6 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ZEROES
2202	T7 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ONES
2406	T10 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ZEROES
2599	T11 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS DUAL ADDRESS TEST
2673	T12 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS DUAL ADDRESS TEST
2792	T13 CACHE ADDRESS MEMORY COMPARATOR TEST
3085	T14 CACHE ADDRESS MEMORY COUNT PATTERN TEST
3299	T15 CACHE ADDRESS MEMORY PARITY LOGIC TEST
3580	T16 CACHE ADDRESS MEMORY DUAL ADDRESS TEST, UPWARD
3877	T17 CACHE ADDRESS MEMORY DUAL ADDRESS TEST, DOWNWARD
4174	T20 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ONES TEST
4312	T21 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ZEROES TEST
4447	T22 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATOB ONES TEST
4609	T23 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATOB ZEROES TEST
4771	T24 CACHE ADDRESS MEMORY POWER UP INVALIDATOR TEST
4853	T25 CACHE DATA MULTIPLEXER, CDMX, TEST
5038	T26 CACHE DATA MEMORY ADDRESS DRIVERS TEST
5155	T27 CACHE DATA MEMORY COUNT PATTERN TEST
5323	T30 CACHE DATA MEMORY PARITY CHECKERS LOW BYTE TEST
5478	T31 CACHE DATA MEMORY PARITY CHECKERS HIGH BYTE TEST
5634	T32 CACHE DATA MEMORY WORST CASE NOISE TEST
5873	T33 CACHE DATA MEMORY CHIP SELECTION LOGIC TEST
6224	T34 CACHE DATA MEMORY BYTE ENABLE LOGIC TEST
6637	T35 CACHE ARBITRATION AND HIGH SPEED I/O TEST
7708	T36 MASS BUS WRITE HIT CYCLE, INVALIDATION TEST
7952	T37 CHECK IVSS, VSIU BITS
7998	T40 CHECK VSIU BIT, WITH IVSS ALREADY SET
8029	T41 CHECK VCIP SETS WHEN CF IS SET
8055	T42 CHECK CACHE FLUSH & VALID STORE SWITCHING
8086	T43 CHECK IVSS INHIBITS SWITCHING OF VALID STORE IN USE
8122	T44 CHECK VALID STORES (A & B) FOR GROUP 0
8241	T45 CHECK VALID STORES (A&B) FOR GROUPS 0 & 1
8325	T46 CHECK VALID STORES (A & B) FOR GROUP 1
8446	T47 CHECK CACHE TURNS OFF WHEN FLUSH IS DONE WITH IVSS SET
8578	T50 CHECK CACHE TURNS OFF ON A BACK-TO-BACK FLUSH
8613	T51 CHECK CACHE-BYPASS
8729	T52 CHECK CACHE IS BYPASSED ON ASRB OPERAND
8787	T53 CHECK CACHE VALID STORE PARITY CHECKER
8905	T54 CHECK THAT CACHE-MISS OCCURS ON A VALID STORE PARITY ERROR
8984	T55 CHECK BYP ON KERNEL PAGE BITS

9021	T56	CHECK BYP ON SUPERVISOR PAGE BITS
9057	T57	CHECK BYP ON USER PAGE BITS
9093	T60	CHECK CACHE BYPASS ON VIRTUAL PAGE
9255		END OF PASS ROUTINE
9299		SCOPE HANDLER ROUTINE
9366		ERROR HANDLER ROUTINE
9417		SAVE AND RESTORE R0-R5 ROUTINES
9464		TYPE ROUTINE
9538		BINARY TO OCTAL (ASCII) AND TYPE
9617		CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
9686		RANDOM NUMBER GENERATOR ROUTINE
9724		TRAP DECODER
9739		TRAP TABLE
9773		POWER DOWN AND UP ROUTINES
9812		ROUTINE TO SIZE MEMORY
9908		DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
10333		SYSTEM DEVICE SIZER
10561		DEVICE HANDLERS
10622		RP04 DISK HANDLER
10782		RS04 DISK HANDLE
10940		RK05 DISK HANDLER
11126		UNIBUS EXERCISER HANDLER
11254		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

```
.TITLE CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2
;*COPYRIGHT (C) 1975, 1978
;*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).
;*
$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
PIRO= 177772   ;;PROGRAM INTERRUPT REQUEST REGISTER
SWR= 177570    ;;SWITCH REGISTER
DISPLAY=SWR
```

```
;*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
```



```

57      000002      R2=      %2          ;;GENERAL REGISTER
58      000003      R3=      %3          ;;GENERAL REGISTER
59      000004      R4=      %4          ;;GENERAL REGISTER
60      000005      R5=      %5          ;;GENERAL REGISTER
61      000006      R6=      %6          ;;GENERAL REGISTER
62      000007      R7=      %7          ;;GENERAL REGISTER
63      .EQUIV      R0,R10      ;;GENERAL REGISTER
64      .EQUIV      R1,R11      ;;GENERAL REGISTER
65      .EQUIV      R2,R12      ;;GENERAL REGISTER
66      .EQUIV      R3,R13      ;;GENERAL REGISTER
67      .EQUIV      R4,R14      ;;GENERAL REGISTER
68      .EQUIV      R5,R15      ;;GENERAL REGISTER
69      000006      SP=%6
70      .EQUIV      SP,KSP      ;;KERNEL STACK POINTER
71      .EQUIV      SP,SSP      ;;SUPERVISOR STACK POINTER
72      .EQUIV      SP,USP      ;;USER STACK POINTER
73      000007      PC=%7
74
75      ;*PRIORITY LEVEL DEFINITIONS
76      000000      PR0=      0          ;;PRIORITY LEVEL 0
77      000040      PR1=      40         ;;PRIORITY LEVEL 1
78      000100      PR2=      100        ;;PRIORITY LEVEL 2
79      000140      PR3=      140        ;;PRIORITY LEVEL 3
80      000200      PR4=      200        ;;PRIORITY LEVEL 4
81      000240      PR5=      240        ;;PRIORITY LEVEL 5
82      000300      PR6=      300        ;;PRIORITY LEVEL 6
83      000340      PR7=      340        ;;PRIORITY LEVEL 7
84
85      ;*'SWITCH REGISTER' SWITCH DEFINITIONS
86      100000      SW15=     100000
87      040000      SW14=     40000
88      020000      SW13=     20000
89      010000      SW12=     10000
90      004000      SW11=     4000
91      002000      SW10=     2000
92      001000      SW09=     1000
93      000400      SW08=     400
94      000200      SW07=     200
95      000100      SW06=     100
96      000040      SW05=     40
97      000020      SW04=     20
98      000010      SW03=     10
99      000004      SW02=     4
100     000002      SW01=     2
101     000001      SW00=     1
102     .EQUIV      SW09,SW9
103     .EQUIV      SW08,SW8
104     .EQUIV      SW07,SW7
105     .EQUIV      SW06,SW6
106     .EQUIV      SW05,SW5
107     .EQUIV      SW04,SW4
108     .EQUIV      SW03,SW3
109     .EQUIV      SW02,SW2
110     .EQUIV      SW01,SW1
111     .EQUIV      SW00,SW0
112

```

```

113      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
114      100000      BIT15= 100000
115      040000      BIT14= 40000
116      020000      BIT13= 20000
117      010000      BIT12= 10000
118      004000      BIT11= 4000
119      002000      BIT10= 2000
120      001000      BIT09= 1000
121      000400      BIT08= 400
122      000200      BIT07= 200
123      000100      BIT06= 100
124      000040      BIT05= 40
125      000020      BIT04= 20
126      000010      BIT03= 10
127      000004      BIT02= 4
128      000002      BIT01= 2
129      000001      BIT00= 1
130      .EQUIV      BIT09,BIT9
131      .EQUIV      BIT08,BIT8
132      .EQUIV      BIT07,BIT7
133      .EQUIV      BIT06,BIT6
134      .EQUIV      BIT05,BIT5
135      .EQUIV      BIT04,BIT4
136      .EQUIV      BIT03,BIT3
137      .EQUIV      BIT02,BIT2
138      .EQUIV      BIT01,BIT1
139      .EQUIV      BIT00,BIT0
140
141      ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
142      000004      ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
143      000010      RESVEC= 10         ;;RESERVED AND ILLEGAL INSTRUCTIONS
144      000014      TBITVEC=14        ;;'T' BIT
145      000014      TRTVEC= 14         ;;TRACE TRAP
146      000014      BPTVEC= 14         ;;BREAKPOINT TRAP (BPT)
147      000020      IOTVEC= 20         ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
148      000024      PWRVEC= 24         ;;POWER FAIL
149      000030      EMTVEC= 30         ;;EMULATOR TRAP (EMT) **ERROR**
150      000034      TRAPVEC=34        ;;'TRAP' TRAP
151      000060      TKVEC= 60          ;;TTY KEYBOARD VECTOR
152      000064      TPVEC= 64          ;;TTY PRINTER VECTOR
153      000114      CACHVEC=114        ;;CACHE ERROR INTERRUPT VECTOR
154      000240      PIRQVEC=240        ;;PROGRAM INTERRUPT REQUEST VECTOR
155      000250      MMVEC= 250         ;;MEMORY MANAGEMENT VECTOR
156
157      .SBTTL      CACHE REGISTER DEFINITIONS
158
159
160      177740      LOADRS = 177740     ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
161      177742      HIADRS = 177742    ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
162      177744      MEMERR = 177744     ;;CACHE ERROR REGISTER
163      177746      CONTRL = 177746    ;;MEMORY CONTROL REGISTER
164      177750      MAINT - 177750     ;;MEMORY MAINTENANCE REGISTER
165      177752      HITMIS = 177752    ;;HIT MISS REGISTER '1' IMPLIES HIT IN CACHE
166
167      .SBTTL      CPU RECISTER DEFINITIONS
168
    
```

169
170
171 177760 SIZELO = 177760 ::MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
172 ::TO GET TO THE LAST 32 WORDS OF MEMORY
173 177762 SIZEHI = 177762 ::HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
174 ::CURRENTLY ALL ZERO
175 177764 SYSTID = 177764 ::SYSTEM ID REGISTER
176 177766 CPUERR = 177766 ::CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
177 ::THE TRAP TO ERRVEC (000004)
178
179

180
181
182 .SBTTL MEMORY MANAGEMENT DEFINITIONS
183

184
185 ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
186

187 177572 MMR0= 177572
188 177574 MMR1= 177574
189 177576 MMR2= 177576
190 172516 MMR3= 172516
191 .EQUIV MMR0,SR0
192 .EQUIV MMR1,SR1
193 .EQUIV MMR2,SR2
194 .EQUIV MMR3,SR3
195

196 ;*USER 'I' PAGE DESCRIPTOR REGISTERS
197

198 177600 UIPDR0= 177600
199 177602 UIPDR1= 177602
200 177604 UIPDR2= 177604
201 177606 UIPDR3= 177606
202 177610 UIPDR4= 177610
203 177612 UIPDR5= 177612
204 177614 UIPDR6= 177614
205 177616 UIPDR7= 177616
206

207 ;*USER 'D' PAGE DESCRIPTOR REGISTERS
208

209 177620 UDPDR0= 177620
210 177622 UDPDR1= 177622
211 177624 UDPDR2= 177624
212 177626 UDPDR3= 177626
213 177630 UDPDR4= 177630
214 177632 UDPDR5= 177632
215 177634 UDPDR6= 177634
216 177636 UDPDR7= 177636
217

218 ;*USER 'I' PAGE ADDRESS REGISTERS
219

220 177640 UIPAR0= 177640
221 177642 UIPAR1= 177642
222 177644 UIPAR2= 177644
223 177646 UIPAR3= 177646
224 177650 UIPAR4= 177650

225 177652 UIPAR5= 177652
226 177654 UIPAR6= 177654
227 177656 UIPAR7= 177656

228
229 ;*USER 'D' PAGE ADDRESS REGISTERS

230
231 177660 UDPAR0= 177660
232 177662 UDPAR1= 177662
233 177664 UDPAR2= 177664
234 177666 UDPAR3= 177666
235 177670 UDPAR4= 177670
236 177672 UDPAR5= 177672
237 177674 UDPAR6= 177674
238 177676 UDPAR7= 177676

239
240 ;*SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS

241
242 172200 SIPDR0= 172200
243 172202 SIPDR1= 172202
244 172204 SIPDR2= 172204
245 172206 SIPDR3= 172206
246 172210 SIPDR4= 172210
247 172212 SIPDR5= 172212
248 172214 SIPDR6= 172214
249 172216 SIPDR7= 172216

250
251 ;*SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS

252
253 172220 SDPDR0= 172220
254 172222 SDPDR1= 172222
255 172224 SDPDR2= 172224
256 172226 SDPDR3= 172226
257 172230 SDPDR4= 172230
258 172232 SDPDR5= 172232
259 172234 SDPDR6= 172234
260 172236 SDPDR7= 172236

261
262 ;*SUPERVISOR 'I' PAGE ADDRESS REGISTERS

263
264 172240 SIPAR0= 172240
265 172242 SIPAR1= 172242
266 172244 SIPAR2= 172244
267 172246 SIPAR3= 172246
268 172250 SIPAR4= 172250
269 172252 SIPAR5= 172252
270 172254 SIPAR6= 172254
271 172256 SIPAR7= 172256

272
273 ;*SUPERVISOR 'D' PAGE ADDRESS REGISTERS

274
275 172260 SDPAR0= 172260
276 172262 SDPAR1= 172262
277 172264 SDPAR2= 172264
278 172266 SDPAR3= 172266
279 172270 SDPAR4= 172270
280 172272 SDPAR5= 172272

281 172274 SDPAR6= 172274
282 172276 SDPAR7= 172276

283
284 ;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS

285
286 172300 KIPDR0= 172300
287 172302 KIPDR1= 172302
288 172304 KIPDR2= 172304
289 172306 KIPDR3= 172306
290 172310 KIPDR4= 172310
291 172312 KIPDR5= 172312
292 172314 KIPDR6= 172314
293 172316 KIPDR7= 172316

294
295 ;*KERNEL 'D' PAGE DESCRIPTOR REGISTERS

296
297 172320 KDPDR0= 172320
298 172322 KDPDR1= 172322
299 172324 KDPDR2= 172324
300 172326 KDPDR3= 172326
301 172330 KDPDR4= 172330
302 172332 KDPDR5= 172332
303 172334 KDPDR6= 172334
304 172336 KDPDR7= 172336

305
306 ;*KERNEL 'I' PAGE ADDRESS REGISTERS

307
308 172340 KIPAR0= 172340
309 172342 KIPAR1= 172342
310 172344 KIPAR2= 172344
311 172346 KIPAR3= 172346
312 172350 KIPAR4= 172350
313 172352 KIPAR5= 172352
314 172354 KIPAR6= 172354
315 172356 KIPAR7= 172356

316
317 ;*KERNEL 'D' PAGE ADDRESS REGISTERS

318
319 172360 KDFAR0= 172360
320 172362 KDFAR1= 172362
321 172364 KDFAR2= 172364
322 172366 KDFAR3= 172366
323 172370 KDFAR4= 172370
324 172372 KDFAR5= 172372
325 172374 KDFAR6= 172374
326 172376 KDFAR7= 172376

327
328
329
330 .SBTTL UNIBUS MAP REGISTER DEFINITIONS

331
332
333
334 ;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
335 ;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'
336

337		
338	170200	MAPL00 = 170200
339	170202	MAPH00 = 170202
340	170204	MAPL01 = 170204
341	170206	MAPH01 = 170206
342	170210	MAPL02 = 170210
343	170212	MAPH02 = 170212
344	170214	MAPL03 = 170214
345	170216	MAPH03 = 170216
346	170220	MAPL04 = 170220
347	170222	MAPH04 = 170222
348	170224	MAPL05 = 170224
349	170226	MAPH05 = 170226
350	170230	MAPL06 = 170230
351	170232	MAPH06 = 170232
352	170234	MAPL07 = 170234
353	170236	MAPH07 = 170236
354	170240	MAPL10 = 170240
355	170242	MAPH10 = 170242
356	170244	MAPL11 = 170244
357	170246	MAPH11 = 170246
358	170250	MAPL12 = 170250
359	170252	MAPH12 = 170252
360	170254	MAPL13 = 170254
361	170256	MAPH13 = 170256
362	170260	MAPL14 = 170260
363	170262	MAPH14 = 170262
364	170264	MAPL15 = 170264
365	170266	MAPH15 = 170266
366	170270	MAPL16 = 170270
367	170272	MAPH16 = 170272
368	170274	MAPL17 = 170274
369	170276	MAPH17 = 170276
370	170300	MAPL20 = 170300
371	170302	MAPH20 = 170302
372	170304	MAPL21 = 170304
373	170306	MAPH21 = 170306
374	170310	MAPL22 = 170310
375	170312	MAPH22 = 170312
376	170314	MAPL23 = 170314
377	170316	MAPH23 = 170316
378	170320	MAPL24 = 170320
379	170320	MAPH24 = 170320
380	170324	MAPL25 = 170324
381	170326	MAPH25 = 170326
382	170330	MAPL26 = 170330
383	170332	MAPH26 = 170332
384	170334	MAPL27 = 170334
385	170336	MAPH27 = 170336
386	170340	MAPL30 = 170340
387	170342	MAPH30 = 170342
388	170344	MAPL31 = 170344
389	170346	MAPH31 = 170346
390	170350	MAPL32 = 170350
391	170352	MAPH32 = 170352
392	170354	MAPL33 = 170354

393	170356	MAPH33 = 170356
394	170360	MAPL34 = 170360
395	170362	MAPH34 = 170362
396	170364	MAPL35 = 170364
397	170366	MAPH35 = 170366
398	170370	MAPL36 = 170370
399	170372	MAPH36 = 170372
400	170374	MAPL37 = 170374
401	170376	MAPH37 = 170376
402		.EQUIV MAPL00,MAPL0
403		.EQUIV MAPH00,MAPH0
404		.EQUIV MAPL01,MAPL1
405		.EQUIV MAPH01,MAPH1
406		.EQUIV MAPL02,MAPL2
407		.EQUIV MAPH02,MAPH2
408		.EQUIV MAPL03,MAPL3
409		.EQUIV MAPH03,MAPH3
410		.EQUIV MAPL04,MAPL4
411		.EQUIV MAPH04,MAPH4
412		.EQUIV MAPL05,MAPL5
413		.EQUIV MAPH05,MAPH5
414		.EQUIV MAPL06,MAPL6
415		.EQUIV MAPH06,MAPH6
416		.EQUIV MAPL07,MAPL7
417		.EQUIV MAPH07,MAPH7

;DEFINITIONS

425	100000	VSPE=BIT15
426	040000	IVSS=BIT14
427	020000	VSIU=BIT13
428	010000	VCIP=BIT12
429	004000	DMMA=BIT11
430	002000	FVPE=BIT10
431	001000	UCB=BIT9
432	000400	FCAC=BIT8
433	000040	S1=BIT5
434	000020	S0=BIT4
435	000010	M1=BIT3
436	000004	M0=BIT2
437	000002	DUT=BIT1
438	000001	DT=BIT0
440	100000	BYP=BIT15
442	000054	S1MOM1=BIT5+BIT3+BIT2
443	000034	S0MOM1=BIT4+BIT3+BIT2
444	000014	MOM1=BIT3+BIT2
446	177746	CONTRL=177746
447	177752	HITMIS=177752
448	177744	MSER=177744


```

449
450
451
452
453
454
455
456
457      000011      TAB=11
458      000044      S1M0=44
459      000030      SOM1=30
460      000054      S1MOM1=54
461      000034      SOMOM1=34
462      000014      M1M0=14
463      000014      MOM1=M1M0
464      140000      TESTR1=140000
465      142000      TESTR2=142000
466      144000      TESTR3=144000
467      001500      STACK=1500
468
469      .SBTTL TRAP CATCHER
470
471      000000      .=0
472      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A '+2,HALT'
473      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
474      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
475
476      .SBTTL STARTING ADDRESS(ES)
477      000200      .-200
478
479 000200 000137 004112      JMP      @#START      ;;JUMP TO STARTING ADDRESS OF PROGRAM
480
481      ;;*****
482
483      .SBTTL          ACT11 HOOKS
484
485      ;*THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11
486      ;*
487      ;*LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOGICAL
488      ;*END OF THE PROGRAM.
489      ;*LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
490      ;*AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS
491      ;*TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:
492      ;*
493      ;*      BIT 15-1 PROGRAM SHOULD BE POWER FAILED WHILE RUNNING
494      ;*      -0 NO POWER FAIL DESIRED
495      ;*
496      ;*      BIT 14=1 PROGRAM RUN TIME IS MEMORY SIZE DEPENDENT
497      ;*      =0 RUN TIME IS NOT MEMORY SIZE DEPENDENT
498      ;*
499      ;*      BITS 13-0 MUST BE ZERO'S
500
501      000204      $SVPC=.      ;;SAVE LOCATION COUNTER
502      000046      .=46      ;;SET LOCATION COUNTER
503 000046 051320      .WORD  $ENDAD      ;;SET LOC.46 TO ADDRESS $ENDAD
504      000052      .-52      ;;SET LOCATION COUNTER
    
```

505 000052 000000
506 000204
507

.WORD 0
.-\$VPC

::SET LOC.52 TO ZERO
::RESTORE LOCATION COUNTER

```

508      ;:*****
509
510      .SBTTL COMMON TAGS
511
512      ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
513      ;*USED IN THE PROGRAM.
514
515      001500      .=1500
516
517      001500      $CMTAG:      ;; START OF COMMON TAGS
518      001500      000000      $PASS: .WORD 0      ;; CONTAINS PASS COUNT
519      001502      000      000      000      $TSTNM: .BYTE 0,0,0      ;; CONTAINS THE TEST NUMBER
520      001505      000      $ERFLG: .BYTE 0      ;; CONTAINS ERROR FLAG
521      001506      000000      $ICNT: .WORD 0      ;; CONTAINS SUBTEST ITERATION COUNT
522      001510      000000      $LPADR: .WORD 0      ;; CONTAINS SCOPE LOOP 1500
523      001512      000000      $LPERR: .WORD 0      ;; CONTAINS SCOPE RETURN FOR ERRORS
524      001514      000000      $ERTTL: .WORD 0      ;; CONTAINS TOTAL ERRORS DETECTED
525      001516      000      $ITEMB: .BYTE 0      ;; CONTAINS ITEM CONTROL BYTE
526      001517      001      $ERMAX: .BYTE 1      ;; CONTAINS MAX. ERRORS PER TEST
527      001520      000000      $ERRPC: .WORD 0      ;; CONTAINS PC OF LAST ERROR INSTRUCTION
528      001522      000000      $GDADR: .WORD 0      ;; CONTAINS 1500 OF 'GOOD' DATA
529      001524      000000      $BDADR: .WORD 0      ;; CONTAINS 1500 OF 'BAD' DATA
530      001526      000000      $GDDAT: .WORD 0      ;; CONTAINS 'GOOD' DATA
531      001530      000000      $BDDAT: .WORD 0      ;; CONTAINS 'BAD' DATA
532      001532      000000      000000      000000      $TKS: .WORD 0,0,0      ;; RESERVED--NOT TO BE USED
533      001540      177560      $TKB: 177560      ;; TTY KBD STATUS
534      001542      177562      $TPS: 177564      ;; TTY KBD BUFFER
535      001544      177564      $TPB: 177566      ;; TTY PRINTER STATUS REG. 1500
536      001546      177566      $NULL: .BYTE 0      ;; TTY PRINTER BUFFER REG. 1500
537      001550      000      $FILLS: .BYTE 2      ;; CONTAINS NULL CHARACTER FOR FILLS
538      001551      002      $FILLC: .BYTE 12      ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
539      001552      012      $TPFLG: .BYTE 0      ;; INSERT FILL CHARS. AFTER A 'LINE FEED'
540      001553      000      $REGAD: .WORD 0      ;; 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
541      001554      000000      $REG0: .WORD 0      ;; CONTAINS THE 1500 FROM
542      001556      000000      $REG1: .WORD 0      ;; WHICH ($REG0) WAS OBTAINED
543      001560      000000      $REG2: .WORD 0      ;; CONTAINS (($REGAD)+0)
544      001562      000000      $REG3: .WORD 0      ;; CONTAINS (($REGAD)+2)
545      001564      000000      $REG4: .WORD 0      ;; CONTAINS (($REGAD)+4)
546      001566      000000      $REG5: .WORD 0      ;; CONTAINS (($REGAD)+6)
547      001570      000000      $REG6: .WORD 0      ;; CONTAINS (($REGAD)+10)
548      001572      000000      $REG7: .WORD 0      ;; CONTAINS (($REGAD)+12)
549      001574      000000      $REG8: .WORD 0      ;; CONTAINS (($REGAD)+14)
550      001576      000000      $REG9: .WORD 0      ;; CONTAINS (($REGAD)+16)
551      001600      000000      $REG10: .WORD 0      ;; CONTAINS (($REGAD)+20)
552      001602      000000      $REG11: .WORD 0      ;; CONTAINS (($REGAD)+22)
553      001604      000000      $REG12: .WORD 0      ;; CONTAINS (($REGAD)+24)
554      001606      000000      $REG13: .WORD 0      ;; CONTAINS (($REGAD)+26)
555      001610      000000      $REG14: .WORD 0      ;; CONTAINS (($REGAD)+30)
556      001612      000000      $REG15: .WORD 0      ;; CONTAINS (($REGAD)+32)
557      001614      000000      $REG16: .WORD 0      ;; CONTAINS (($REGAD)+34)
558      001616      000000      $REG17: .WORD 0      ;; CONTAINS (($REGAD)+36)
559      001620      000000      $REG20: .WORD 0      ;; CONTAINS (($REGAD)+40)
560      001622      000000      $REG21: .WORD 0      ;; CONTAINS (($REGAD)+42)
561      001624      000000      $REG22: .WORD 0      ;; CONTAINS (($REGAD)+44)
562      001626      000000      $REG23: .WORD 0      ;; CONTAINS (($REGAD)+46)
563      001628      000000      $TMP0: .WORD 0      ;; USER DEFINED
    
```

```

564 001630 000000 $TMP1: .WORD 0 ;;USER DEFINED
565 001632 000000 $TMP2: .WORD 0 ;;USER DEFINED
566 001634 000000 $TMP3: .WORD 0 ;;USER DEFINED
567 001636 000000 $TMP4: .WORD 0 ;;USER DEFINED
568 001640 000000 $TMP5: .WORD 0 ;;USER DEFINED
569 001642 000000 $TMP6: .WORD 0 ;;USER DEFINED
570 001644 000C00 $TMP7: .WORD 0 ;;USER DEFINED
571 001646 000000 $TMP10: .WORD 0 ;;USER DEFINED
572 001650 000000 $TMP11: .WORD 0 ;;USER DEFINED
573 001652 000000 $TMP12: .WORD 0 ;;USER DEFINED
574 001654 000000 $TMP13: .WORD 0 ;;USER DEFINED
575 001656 000000 $TMP14: .WORD 0 ;;USER DEFINED
576 001660 000000 $TMP15: .WORD 0 ;;USER DEFINED
577 001662 000000 $TMP16: .WORD 0 ;;USER DEFINED
578 001664 000000 $TMP17: .WORD 0 ;;USER DEFINED
579 001666 000000 $TMP20: .WORD 0 ;;USER DEFINED
580 001670 000000 $TMP21: .WORD 0 ;;USER DEFINED
581 001672 000000 $TMP22: .WORD 0 ;;USER DEFINED
582 001674 000000 $TMP23: .WORD 0 ;;USER DEFINED
583 001676 000000 $TIMES: 0 ;;MAX. NUMBER OF ITERATIONS
584 001700 000000 $ESCAPE: 0 ;;ESCAPE ON ERROR 1500
585 001702 177607 000377 $BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
586 001706 077 $QUES: .ASCII /?/ ;;QUESTION MARK
587 001707 015 $CRLF: .ASCII <15> ;;CARRIAGE RETURN
588 001710 000012 $LF: .ASCIZ <12> ;;LINE FEED
589 001712 000 KB11E: .BYTE 0 ;:1174 WITHOUT MP CACHE FLAG
590 001713 000 KB11EM: .BYTE 0 ;:1174 WITH MP CACHE FLAG
591 001714 000 KB11CM: .BYTE 0 ;:KB11CM FLAG (1170 WITH MP MODS)
592 001715 000 CISP: .BYTE 0 ;:CISP OPTION PRESENT FLAG
593
594 ;OPCODE FOR MFPT INSTRUCTION (AVAILABLE ON KB11-E AND KB11-EM ONLY)
595 000007 MFPT-7
    
```

```

596      ::*****
597
598      .SBTTL  ERROR POINTER TABLE
599
600      :*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
601      :*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
602      :*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
603      :*NOTE1:      IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
604      :*NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
605
606      :*      EM      ::POINTS TO THE ERROR MESSAGE
607      :*      DH      ::POINTS TO THE DATA HEADER
608      :*      DT      ::POINTS TO THE DATA
609      :*      DF      ::POINTS TO THE DATA FORMAT
610
611
612      001716      $ERRTB:
613
614
615      :ERROR TABLE FOR ERROR TYPE OUT:
616      :ITEM 1
617      001716      070420      107657      113442      .WORD      EM1,DH1,DT1,DF1
618      001724      113117
619      :ITEM 2
620      001726      070505      107732      113454      .WORD      EM2,DH2,DT2,DF2
621      001734      113123
622      :ITEM 3
623      001736      070677      107732      113454      .WORD      EM3,DH3,DT3,DF3
624      001744      113123
625      :ITEM 4
626      001746      071013      107732      113454      .WORD      EM4,DH4,DT4,DF4
627      001754      113123
628      :ITEM 5
629      001756      071126      110033      113472      .WORD      EM5,DH5,DT5,DF5
630      001764      113131
631      :ITEM 6
632      001766      071206      110033      113472      .WORD      EM6,DH6,DT6,DF6
633      001774      113131
634      :ITEM 7
635      001776      071266      110033      113472      .WORD      EM7,DH7,DT7,DF7
636      002004      113131
637      :ITEM 10
638      002006      071360      110033      113472      .WORD      EM10,DH10,DT10,DF10
639      002014      113131
640      :ITEM 11
641      002016      071451      110065      113514      .WORD      EM11,DH11,DT11,DF11
642      002024      113141
643      :ITEM 12
644      002026      071537      110141      113540      .WORD      EM12,DH12,DT12,DF12
645      002034      113152
646      :ITEM 13
647      002036      071664      110234      113556      .WORD      EM13,DH13,DT13,DF13
648      002044      113160
649      :ITEM 14
650      002046      071744      110307      113572      .WORD      EM14,DH14,DT14,DF14
651      002054      113165
    
```

652					:ITEM 15		
653	002056	072003	110402	113606	.WORD	EM15,DH15,DT15,DF15	
654	002064	113172					
655					:ITEM 16		
656	002066	072053	110426	113614	.WORD	EM16,DH16,DT16,DF16	
657	002074	113174					
658					:ITEM 17		
659	002076	072127	110514	113632	.WORD	EM17,DH17,DT17,DF17	
660	002104	113202					
661					:ITEM 20		
662	002106	072127	110514	113714	.WORD	EM20,DH20,DT20,DF20	
663	002114	113202					
664					:ITEM 21		
665	002116	072210	110574	113776	.WORD	EM21,DH21,DT21,DF21	
666	002124	113232					
667					:ITEM 22		
668	002126	072274	110647	114054	.WORD	EM22,DH22,DT22,DF22	
669	002134	113260					
670					:ITEM 23		
671	002136	072510	110716	114064	.WORD	EM23,DH23,DT23,DF23	
672	002144	113263					
673					:ITEM 24		
674	002146	072274	110647	114100	.WORD	EM24,DH24,DT24,DF24	
675	002154	113260					
676					:ITEM 25		
677	002156	072510	110716	114110	.WORD	EM25,DH25,DT25,DF25	
678	002164	113263					
679					:ITEM 26		
680	002166	072644	111006	114124	.WORD	EM26,DH26,DT26,DF26	
681	002174	113270					
682					:ITEM 27		
683	002176	073011	111053	114136	.WORD	EM27,DH27,DT27,DF27	
684	002204	113274					
685					:ITEM 30		
686	002206	072274	110647	114152	.WORD	EM30,DH30,DT30,DF30	
687	002214	113260					
688					:ITEM 31		
689	002216	073156	110716	114162	.WORD	EM31,DH31,DT31,DF31	
690	002224	113263					
691					:ITEM 32		
692	002226	072274	110647	114176	.WORD	EM32,DH32,DT32,DF32	
693	002234	113260					
694					:ITEM 33		
695	002236	073156	110716	114206	.WORD	EM33,DH33,DT33,DF33	
696	002244	113263					
697					:ITEM 34		
698	002246	073315	111145	114222	.WORD	EM34,DH34,DT34,DF34	
699	002254	113301					
700					:ITEM 35		
701	002256	073421	111145	114222	.WORD	EM35,DH35,DT35,DF35	
702	002264	113301					
703					:ITEM 36		
704	002266	073530	111225	114236	.WORD	EM36,DH36,DT36,DF36	
705	002274	113306					
706					:ITEM 37		
707	002276	073662	111272	114250	.WORD	EM37,DH37,DT37,DF37	

708	002304	113312			
709					:ITEM 40
710	002306	073744	111424	114276	.WORD EM40,DH40,DT40,DF40
711	002314	113324			
712					:ITEM 41
713	002316	074117	111357	114264	.WORD EM41,DH41,DT41,DF41
714	002324	113320			
715					:ITEM 42
716	002326	074303	111357	114264	.WORD EM42,DH42,DT42,DF42
717	002334	113320			
718					:ITEM 43
719	002336	074556	111424	114276	.WORD EM43,DH43,DT43,DF43
720	002344	113324			
721					:ITEM 44
722	002346	074704	111473	114320	.WORD EM44,DH44,DT44,DF44
723	002354	113334			
724					:ITEM 45
725	002356	075100	111473	114320	.WORD EM45,DH45,DT45,DF45
726	002364	113334			
727					:ITEM 46
728	002366	075277	111566	114360	.WORD EM46,DH46,DT46,DF46
729	002374	113353			
730					:ITEM 47
731	002376	075420	111566	114360	.WORD EM47,DH47,DT47,DF47
732	002404	113353			
733					:ITEM 50
734	002406	074704	111473	114412	.WORD EM50,DH50,DT50,DF50
735	002414	113334			
736					:ITEM 51
737	002416	075100	111473	114412	.WORD EM51,DH51,DT51,DF51
738	002424	113334			
739					:ITEM 52
740	002426	075277	111566	114452	.WORD EM52,DH52,DT52,DF52
741	002434	113353			
742					:ITEM 53
743	002436	075420	111566	114452	.WORD EM53,DH53,DT53,DF53
744	002444	113353			
745					:ITEM 54
746	002446	075544	111612	114504	.WORD EM54,DH54,DT54,DF54
747	002454	113367			
748					
749					:ITEM 55
750					
751	002456	075717			EM55
752	002460	111653			DH55
753	002462	114514			DT55
754	002464	000000			0
755					
756					
757					
758					:ITEM 56
759					
760	002466	075750			EM56
761	002470	111653			DH55
762	002472	114514			DT55
763	002474	000000			0

764			
765			
766			:ITEM 57
767			
768	002476	076016	EM57
769	002500	111653	DH55
770	002502	114514	DT55
771	002504	113372	DF57
772			
773			
774			:ITEM 60
775			
776	002506	076057	EM60
777	002510	111653	DH55
778	002512	114514	DT55
779	002514	113372	DF61
780			
781			:ITEM 61
782			
783	002516	076116	EM61
784	002520	111653	DH55
785	002522	114514	DT55
786	002524	113372	DF61
787			
788			
789			:ITEM 62
790			
791	002526	076205	EM62
792	002530	111653	DH55
793	002532	114514	DT55
794	002534	113372	DF62
795			
796			
797			:ITEM 63
798			
799	002536	076237	EM63
800	002540	111653	DH55
801	002542	114514	DT55
802	002544	113372	DF63
803			
804			
805			:ITEM 64
806			
807	002546	076315	EM64
808	002550	111653	DH55
809	002552	114514	DT55
810	002554	113372	DF64
811			
812			
813			:ITEM 65
814			
815	002556	076376	EM65
816	002560	111653	DH55
817	002562	114514	DT55
818	002564	113372	DF65
819			

820			
821			:ITEM 66
822			
823	002566	076464	EM66
824	002570	111667	DH66
825	002572	114522	DT66
826	002574	113374	DF66
827			
828			
829			:ITEM 67
830			
831	002576	076574	EM67
832	002600	111667	DH66
833	002602	114522	DT66
834	002604	113374	DF67
835			
836			
837			:ITEM 70
838			
839	002606	076707	EM70
840	002610	111667	DH66
841	002612	114522	DT66
842	002614	113374	DF70
843			
844			:ITEM 71
845			
846	002616	077023	EM71
847	002620	111733	DH71
848	002622	114522	DT66
849	002624	113374	DF71
850			
851			:ITEM 72
852			
853	002626	077071	EM72
854	002630	111667	DH66
855	002632	114522	DT66
856	002634	113374	DF72
857			
858			:ITEM 73
859			
860	002636	077220	EM73
861	002640	111667	DH66
862	002642	114522	DT66
863	002644	113374	DF73
864			
865			:ITEM 74
866			
867	002646	077330	EM74
868	002650	111667	DH66
869	002652	114522	DT66
870	002654	113374	DF74
871			
872			:ITEM 75
873			
874	002656	077432	EM75
875	002660	111667	DH66

876	002662	114522					DT66
877	002664	113374					DF75
878							
879						:ITEM	76
880							
881	002666	077546					EM76
882	002670	111667					DH66
883	002672	114522					DT66
884	002674	113374					DF76
885							
886						:ITEM	77
887							
888	002676	077657					EM77
889	002700	111667					DH66
890	002702	114522					DT66
891	002704	113374					DF77
892							
893						:ITEM	0
894							
895	002706	000000	000000	000000		.WORD	0,0,0,0
896	002714	000000					
897							
898						:ITEM	0
899							
900	002716	000000	000000	000000		.WORD	0,0,0,0
901	002724	000000					
902							
903						:ITEM	0
904							
905	002726	000000	000000	000000		.WORD	0,0,0,0
906	002734	000000					
907							
908						:ITEM	103
909	002736	100122				EM103	:NO PARITY ERROR TRAP ON VALID STORE PARITY ERROR
910	002740	111653				DH55	
911	002742	114514				DT55	
912	002744	113372				DF103	
913							
914						:ITEM	104
915							
916	002746	100203				EM104	:TEST-DATA-REFERENCE GIVING VALID STORE PARITY
917	002750	111653				DH55	:ERROR WAS NOT A MISS
918	002752	114514				DT55	
919	002754	113372				DF104	
920							
921						:ITEM	105
922							
923	002756	100307				EM105	:FVPE DID NOT GET CLEARED AFTER VSPE OCCURED
924	002760	111653				DH55	
925	002762	114514				DT55	
926	002764	113372				DF105	
927							
928						:ITEM	106
929							
930	002766	100363				EM106	:VALID-STORE-PARITY-ERROR BIT DID NOT SET IN CCR ONVSPE
931	002770	111653				DH55	

```

932 002772 114514 DT55
933 002774 113372 DF106
934
935 ;ITEM 107
936
937 002776 100453 EM107 ;FAST ADDRESS MEMORY PARITY ERROR BITS (4,S) NOT
938 003000 111765 DH107 ;SET CORRECTLY IN MSER ON VSPE
939 003002 114552 DT107
940 003004 113406 DF107
941
942 ;ITEM 110
943
944 003006 100572 EM110 ;VSIV SWITCHED ON VSPE
945 003010 111653 DH55
946 003012 114514 DT55
947 003014 113372 DF110
948
949 ;ITEM 111
950
951 003016 100620 EM111 ;MEMORY SYSTEM ERROR REGISTER COULD NOT BE CLEARED
952 003020 112037 DH111
953 003022 114514 DT55
954 003024 113372 DF111
955
956 ;ITEM 112
957
958 003026 100702 EM112 ;VSPE COULD NOT BE CLEARED IN CCR
959 003030 111653 DH55
960 003032 114514 DT55
961 003034 113372 DF112
962
963 ;ITEM 113
964
965 003036 100743 EM113 ;TEST-DATA-REFERENCE NOT A HIT
966 003040 111667 DH66
967 003042 114522 DT66
968 003044 113374 DF113
969
970 ;ITEM 0
971
972 003046 000000 000000 000000 .WORD 0,0,0,0
973 003054 000000
974
975 ;ITEM 115
976
977 003056 101001 EM115 ;TEST DATA REFERENCE NOT A MISS
978 003060 112054 DH115 ;CACHE DID NOT TURN OFF ON BACK-TO-RACK FLUSH
979 003062 114514 DT55
980 003064 113372 DF115
981
982 ;ITEM 116
983
984 003066 000000 000000 000000 .WORD 0,0,0,0
985 003074 000000
986
987 ;ITEM 117
    
```

988						
989	003076	000000	000000	000000	.WORD	0,0,0,0
990	003104	000000				
991						
992					:ITEM	120
993						
994	003106	000000	000000	000000	.WORD	0,0,0,0
995	003114	000000				
996						
997					:ITEM	121
998						
999	003116	000000	000000	000000	.WORD	0,0,0,0
1000	003124	000000				
1001						
1002					:ITEM	122
1003						
1004	003126	000000	000000	000000	.WORD	0,0,0,0
1005	003134	000000				
1006						
1007					:ITEM	123
1008	003136	101116			EM123	:BYP BIT IN KIPDR COULD NOT BE CLEARED
1009	003140	112073			DH123	: PC KIPDR (KIPDR)
1010	003142	114566			DT123	: \$ERRPC,\$REG0,\$REG1,0
1011	003144	113413			DF123	:0,0,0
1012					:ITEM	124
1013	003146	101164			EM124	:BYP BIT IN KIPDR COULD NOT BE SET
1014	003150	112073			DH123	
1015	003152	114566			DT123	
1016	003154	113413			DF123	
1017						
1018					:ITEM	125
1019	003156	101226			EM125	:TEST DATA COULD NOT BE MADE HIT
1020	003160	112122			DH125	: PC CCR PARADR PAR PDR TST-DATA-ADR
1021	003162	114576			DT125	: \$ERRPC,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4,0
1022	003164	113400			DF100	
1023						
1024					:ITEM	126
1025	003166	101266			EM126	:TEST DATA REFERENCE NOT A MISS
1026						:CACHED DATA WAS NOT FORCED A MISS ON VIRTUAL PAGE BYPASS
1027	003170	112122			DH125	
1028	003172	114576			DT125	
1029	003174	113400			DF100	
1030						
1031					:ITEM	127
1032	003176	101417			EM127	:TEST DATA REFERENCE NOT A MISS
1033						:CACHED DATA WAS NOT INVALIDATED ON VIRTUAL BYPASS
1034	003200	112122			DH125	
1035	003202	114576			DT125	
1036	003204	113400			DF100	
1037					:ITEM	130
1038	003206	101546			EM130	:BYP BIT IN SIPDR COULD NOT BE CLEARED
1039	003210	112211			DH130	: PC SIPDR (SIPDR)
1040	003212	114566			DT123	
1041	003214	113413			DF123	
1042						
1043					:ITEM	131

1100	003416	106061	112463	114652	.WORD	EM151,DH151,DT151,DF151
1101	003424	113432				
1102					:ITEM 152	
1103	003426	106061	112532	114652	.WORD	EM152,DH152,DT152,DF152
1104	003434	113432				
1105					:ITEM 153	
1106	003436	106061	112601	114652	.WORD	EM153,DH153,DT153,DF153
1107	003444	113432				
1108					:ITEM 154	
1109	003446	106142	112663	114662	.WORD	EM154,DH154,DT154,DF154
1110	003454	113435				
1111					:ITEM 155	
1112	003456	106174	112721	114662	.WORD	EM155,DH155,DT155,DF155
1113	003464	113435				
1114					:ITEM 156	
1115	003466	106226	112757	114662	.WORD	EM156,DH156,DT156,DF156
1116	003474	113435				
1117					:ITEM 0	
1118	003476	000000	000000	000000	.WORD	0,0,0,0
1119	003504	000000				
1120					:ITEM 160	
1121	003506	106273	113015	114652	.WORD	EM160,DH160,DT160,DF160
1122	003514	113435				
1123					:ITEM 161	
1124	003516	106325	113043	114652	.WORD	EM161,DH161,DT161,DF161
1125	003524	113435				
1126						
1127					:ITEM 162	
1128	003526	106373	113073	114674	.WORD	EM162,DH162,DT162,DF55
1129	003534	113372				
1130						
1131					:ITEM 163	
1132	003536	106603	113073	114674	.WORD	EM163,DH162,DT162,DF55
1133	003544	113372				
1134						
1135					:ITEM 164	
1136	003546	106675	113073	114674	.WORD	EM164,DH162,DT162,DF55
1137	003554	113372				
1138						
1139					:ITEM 165	
1140	003556	106754	113073	114674	.WORD	EM165,DH162,DT162,DF55
1141	003564	113372				
1142						
1143					:ITEM 166	
1144	003566	106773	113073	114674	.WORD	EM166,DH162,DT162,DF55
1145	003574	113372				
1146						
1147					:ITEM 167	
1148	003576	107057	113073	114674	.WORD	EM167,DH162,DT162,DF55
1149	003604	113372				
1150						
1151					:ITEM 170	
1152	003606	107163	113073	114674	.WORD	EM170,DH162,DT162,DF55
1153	003614	113372				
1154						
1155					:ITEM 171	

1156	003616	107226	113073	114674	.WORD	EM171,DH162,DT162,DF55
1157	003624	113372				
1158						
1159					:ITEM 172	
1160	003626	107272	113073	114674	.WORD	EM172,DH162,DT162,DF55
1161	003634	113372				
1162						
1163					:ITEM 173	
1164	003636	107356	113073	114674	.WORD	EM173,DH162,DT162,DF55
1165	003644	113372				
1166						
1167					:ITEM 174	
1168	003646	107544	113073	114674	.WORD	EM174,DH162,DT162,DF55
1169	003654	113372				
1170						
1171					:ITEM 175	
1172	003656	107607	113073	114674	.WORD	EM175,DH162,DT162,DF55
1173	003664	113372				
1174	003666	000016			RS4REG: .WORD	16
1175	003670	172040			RS4CS1: .WORD	172040
1176	003672	000000			RS4WC: .WORD	0
1177	003674	000000			RS4BA: .WORD	0
1178	003676	000000			RS4DA: .WORD	0
1179	003700	000000			RS4CS2: .WORD	0
1180	003702	000000			RS4DS: .WORD	0
1181	003704	000000			RS4ER: .WORD	0
1182	003706	000000			RS4AS: .WORD	0
1183	003710	000000			RS4LA: .WORD	0
1184	003712	000000			RS4DB: .WORD	0
1185	003714	000000			RS4MR: .WORD	0
1186	003716	000000			RS4DT: .WORD	0
1187	003720	000000			RS4BAE: .WORD	0
1188	003722	000000			RS4CS3: .WORD	0
1189						
1190	003724	000026			RP4REG: .WORD	26
1 91	003726	176700			RP4CS1: .WORD	176700
1192	003730	000000			RP4WC: .WORD	0
1193	003732	000000			RP4BA: .WORD	0
1194	003734	000000			RP4DA: .WORD	0
1195	003736	000000			RP4CS2: .WORD	0
1196	003740	000000			RP4DS: .WORD	0
1197	003742	000000			RP4RR1: .WORD	0
1198	003744	000000			RP4AS: .WORD	0
1199	003746	000000			RP4LA: .WORD	0
1200	003750	000000			RP4DB: .WORD	0
1201	003752	000000			RP4MR: .WORD	0
1202	003754	000000			RP4DT: .WORD	0
1203	003756	000000			RP4SN: .WORD	0
1204	003760	000000			RP4OF: .WORD	0
1205	003762	000000			RP4DC: .WORD	0
1206	003764	000000			RP4CCC: .WORD	0
1207	003766	000000			RP4RR2: .WORD	0
1208	003770	000000			RP4RR3: .WORD	0
1209	003772	000000			RP4EC1: .WORD	0
1210	003774	000000			RP4EC2: .WORD	0
1211	003776	000000			RP4BAE: .WORD	0

1212	004000	000000		RP4CS3: .WORD	0
1213					
1214	004002	000014		RH4REG: .WORD	14
1215	004004	160100		RH4CS1: .WORD	160100
1216	004006	000000		RH4WC: .WORD	0
1217	004010	000000		RH4BA: .WORD	0
1218	004012	000000		RH4MR2: .WORD	0
1219	004014	000000		RH4CS2: .WORD	0
1220	004016	000000		RH4ST: .WORD	0
1221	004020	000000		RH4ER: .WORD	0
1222	004022	000000		RH4AS: .WORD	0
1223	004024	000000		RH4DR: .WORD	C
1224	004026	000000		RH4DB: .WORD	0
1225	004030	000000		RH4MR1: .WORD	0
1226	004032	000000		RH4DT: .WORD	0
1227					
1228	004034	000002		RH4REX: .WORD	2
1229	004036	160174		RH4AE: .WORD	160174
1230	004040	000000		RH4CS3: .WORD	0
1231					
1232	004042	000007		RK5REG: .WORD	7
1233	004044	177400		RK5DS: .WORD	177400
1234	004046	000000		RK5ER: .WORD	0
1235	004050	000000		RK5CS1: .WORD	0
1236	004052	000000		RK5WC: .WORD	0
1237	004054	000000		RK5BA: .WORD	0
1238	004056	000000		RK5DA: .WORD	0
1239	004060	000000		RK5DB: .WORD	0
1240					
1241					
1242	004062	000006		UBEREG: .WORD	6
1243	004064	170000		UBEDB: .WORD	170000
1244	004066	000000		UBECC: .WORD	0
1245	004070	000000		UBEBA: .WORD	0
1246	004072	000000		UBECR1: .WORD	0
1247	004074	000000		UBECLR: .WORD	0
1248	004076	000000		UBECR2: .WORD	0
1249					
1250				;THESE ARE THE DEVICE TRAP VECTOR ADDRESSES:	
1251	004100	000204		RS4V: .WORD	204
1252	004102	000254		RP4V: .WORD	254
1253	004104	000774		RH4V: .WORD	774
1254	004106	000220		RK5V: .WORD	220
1255	004110	000510		UBEV: .WORD	510
1256					
1257					
1258					
1259	004112	005037	001502	START: CLR	\$STNM
1260	004116	012737	000340	MOV	#340,@MPS ;:LOCK OUT ALL INTERRUPTS
1261	004124	012706	001500	MOV	#\$CMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
1262	004130	005026		CLR	(R6)+ ;:CLEAR MEMORY LOCATION
1263	004132	022706	001540	CMP	#\$TKS,R6 ;:DONE?
1264	004136	001374		BNE	.-6 ;:LOOP BACK IF NO
1265	004140	012706	001500	MOV	#\$STACK,SP ;:SETUP THE STACK POINTER
1266	004144	012737	051354	MOV	#\$SCOPE,@#IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
1267	004152	012737	000340	MOV	#340,@#IOTVEC+2 ;:LEVEL 7

```

1268 004160 012737 051636 000030      MOV      # $ERROR,@EMTVEC  ;; EMT VECTOR FOR ERROR ROUTINE
1269 004166 012737 000340 000032      MOV      #340,@EMTVEC+2  ;; LEVEL 7
1270 004174 012737 053112 000034      MOV      # $TRAP,@TRAPVEC  ;; TRAP VECTOR FOR TRAP CALLS
1271 004202 012737 000340 000036      MOV      #340,@TRAPVEC+2  ;; LEVEL 7
1272 004210 012737 053206 000024      MOV      # $PWRDN,@PWRVEC  ;; POWER FAILURE VECTOR
1273 004216 012737 000340 000026      MOV      #340,@PWRVEC+2  ;; LEVEL 7
1274 004224 013737 051250 051242      MOV      $ENDCT,$EOPCT    ;; SETUP END-OF-PROGRAM COUNTER
1275 004232 005037 001676                CLR      $TIMES           ;; INITIALIZE NUMBER OF ITERATIONS
1276 004236 005037 001700                CLR      $ESCAPE         ;; CLEAR THE ESCAPE ON ERROR ADDRESS
1277 004242 112737 000001 001517      MOV     B #1,$SERMAX      ;; ALLOW ONE ERROR PER TEST
1278 004250 012737 004250 001510      MOV      #.,$LPADR       ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
1279 004256 012737 004256 001512      MOV      #.,$LPERR      ;; SETUP THE ERROR LOOP ADDRESS
1280 004264 005227 177777                INC      #-1             ;; FIRST TIME?
1281 004270 001044                BNE     64$             ;; BRANCH IF NO
1282 004272 022737 051320 000042      CMP      # $ENDAD,@#42   ;; ACT-11?
1283 004300 001440                BEQ     64$             ;; BRANCH IF YES
1284 004302 104400 004310                TYPE    ,65$           ;; TYPE ASCIZ STRING
1285 004306 000435                BR      64$             ;; GET OVER THE ASCIZ
1286                                ;;65$: .ASCIZ <CRLF>'CEKBD-D PDP 11/70-74MP CACHE MEMORY DIAGNOSTIC PART 2 '<CRLF>
1287                                64$:
1288 004402 005227 177777                INC      #-1             ;; FIRST TIME?
1289 004406 001043                BNE     66$             ;; BRANCH IF NO
1290 004410 022737 051320 000042      CMP      # $ENDAD,@#42   ;; ACT-11?
1291 004416 001437                BEQ     66$             ;; BRANCH IF YES
1292 004420 104400 004426                TYPE    ,67$           ;; TYPE ASCIZ STRING
1293 004424 000434                BR      66$             ;; GET OVER THE ASCIZ
1294                                ;;67$: .ASCIZ <CRLF>'PROGRAMMABLE RP04 DRIVES WILL NOT BE USED BY TEST 35'<CRLF>
1295                                66$:
1296                                ;;*****
1297
1298                                ; SIZE MEMORY AND COMPARE IT WITH THE SYSTEM SIZE REGISTER
1299                                ;PRINT A WARNING MESSAGE IF THEY DISAGREE.
1300
1301 004516 052737 000200 053410      BIS      #BIT07,$KT11
1302 004524 004737 053342      JSR     PC,$SIZE
1303 004530 062737 000037 053726      ADD      #37,$LSTBK      ;; ADJUST THE SIZE FOR COMPARISON
1304                                ;; TO SIZE REGISTER
1305 004536 023737 177760 053726      CMP      @#SIZELO,$LSTBK ;; IS THE ACTUAL SIZE REFLECTED BY THE
1306                                ;; SIZE REGISTER?
1307 004544 001420                BEQ     OKSIZ
1308 004546 104400 070007      TYPE    ,MS01
1309 004552 104400 070144      TYPE    ,MS02
1310 004556 104400 001707      TYPE    , $CRLF
1311 004562 013746 177760      MOV     @#SIZELO,-(SP)   ;; SAVE @#SIZELO FOR TYPEOUT
1312 004566 104404                TYPOS   ;; GO TYPE--OCTAL ASCII
1313 004570                .BYTE  6                ;; TYPE 6 DIGIT(S)
1314 004571                .BYTE  0                ;; SUPPRESS LEADING ZEROS
1315 004572 104400 070173      TYPE    ,MS03
1316 004576 013746 053726      MOV     $LSTBK,-(SP)    ;; SAVE $LSTBK FOR TYPEOUT
1317 004602 104404                TYPOS   ;; GO TYPE--OCTAL ASCII
1318 004604                .BYTE  6                ;; TYPE 6 DIGIT(S)
1319 004605                .BYTE  0                ;; SUPPRESS LEADING ZEROS
1320 004606      OKSIZ:
1321
1322                                ;;
1323                                ;;*** TEST FOR VARIOUS KB11 PROCESSORS ***
    
```



```

1380 005036 012737 000400 001712      MOV      #BIT8,@#KB11E      ;SET UPPER BYTE (KB11-EM)
1381 005044 000402                      BR       2$                ;DONE
1382 005046 105237 001714      1$: INCB  @#KB11CM          ;YES, FLAG THIS AS A MODIFIED PROCESSOR
1383 005052 000403      2$: BR       ENDKB          ;DONE DETERMINING WHICH CPU
1384
1385 005054                      MFPTTR:                      ;HERE IF MFPT TRAPPED. SEE IF 1170 OR KB11CM
1386 005054 012716 004634      MOV      #T1,(SP)          ;SET UP RETURN ADDRESS FOR RTI
1387 005060 000002                      RTI                          ;RETURN
1388 005062                      ENDKB:
1389 005062 005227 177777      INC      #-1                ;FIRST TIME?
1390 005066 001026      BNE     100$                ;BR IF NO
1391 005070 104400 070203      TYPE    ,MSG1              ;<15><12>CPU UNDER TEST FOUND TO BE A
1392 005074 005737 001712      TST     @#KB11E            ;IS THIS A KB11-E OR KB11-EM?
1393 005100 001011      BNE     101$                ;BR IF EITHER ONE
1394 005102 105737 001714      TSTB   @#KB11CM           ;IS IT A 11/74 (KB11CM)
1395 005106 001003      BNE     1$                  ;BR IF IT IS
1396 005110 104400 070253      TYPE    ,MSG3              ;KB11-B/C<15><12>
1397 005114 000413      BR      100$                ;SKIP OTHER MESSAGE
1398 005116 104400 070265      1$: TYPE    ,MSG4              ;11/74 (KB11CM)<15><12>
1399 005122 000410      BR      100$                ;SKIP CISP MESSAGE
1400 005124 105737 001712      101$: TSTB   @#KB11E            ;IS IT A KB11-E?
1401 005130 001403      BEQ     102$                ;BR IF NOT. MUST BE KB11-EM
1402 005132 104400 070316      TYPE    ,MSG5              ;KB11-E<15><12>
1403 005136 000402      BR      100$                ;SKIP KB11-EM MESSAGE
1404 005140 104400 070242      102$: TYPE    ,MSG2              ;KB11-EM<15><12>
1405 005144      100$:
1406
1407      ;THIS ROUTINE SAVES THE TOP 1500 (DEC) WORDS OF THE FIRST 28K OF
1408      ;MEMORY. THESE LOCATIONS SHOULD CONTAIN EITHER THE MONITOR OR THE
1409      ;LOADER WHICH LOADED THE PROGRAM. NOTE THAT TO RESTORE THIS PART
1410      ;OF CORE, THAT IS TO RESTORE THE LOADER OR MONITOR, ALL THE USER
1411      ;MUST DO IS TYPE ^C (CONTROL-C), WHILE THIS PROGRAM IS RUNNING.
1412      ;THIS WILL AUTOMATICALLY RESTORE THE TOP PART OF MEMORY TO ITS STATE
1413      ;BEFORE THIS PROGRAM WAS STARTED! AFTER THE MONITOR (OR LOADER) HAS BEEN
1414      ;RESTORED THIS PROGRAM WILL HALT.
1415 005144 005237 054644      LOOP: INC      MONF          ;INCREMENT THE FLAG WHICH INDICATES
1416 005150 001013      BNE     TOP                ;WHETHER OR NOT THE TOP OF MEMORY
1417      ;IN THE FIRST 28K HAS BEEN SAVED.
1418 005152 013737 000060 054642      MOV      @#TKVEC,MONTTY    ;SAVE THE INITIAL CONTENTS OF THE TTY
1419      ;KEYBOARD INTERRUPT VECTOR.
1420 005160 012700 002734      MOV      #^D1500,R0        ;IF NOT THEN SAVE IT.
1421 005164 012701 116710      MOV      #BOTTOM+4,R1      ;SAVE IT AT THE BOTTOM OF THIS PROGRAM.
1422 005170 012702 160000      MOV      #160000,R2        ;GET THE ADDRESS OF THE END OF THE MONITOR.
1423 005174 014221      1$: MOV      -(R2),(R1)+    ;SAVE 1500 (DEC) LOCATIONS (WORDS)
1424 005176 077002      SOB     R0,1$
1425 005200 012737 000044 177770      TOP: MOV      #44,@#177770
1426
1427 005206 012737 054512 000060      MOV      #RESMON,@#TKVEC   ;SET THE KEYBOARD INTERRUPT VECTOR.
1428 005214 012737 000340 000062      MOV      #340,@#TKVEC+2
1429 005222 005077 174314      CLR     @#TKB              ;MAKE SURE THE KEYBOARD BUFFER IS CLEAR.
1430 005226 152777 000100 174304      BISB   #BIT6,@#STKS        ;TURN ON INTERRUPT ENABLE FOR THE KEYBOARD.
1431 005234 012737 054050 000004      MOV      #CPSPUR,@#4       ;SET UP FOR UNEXPECTED ERRORS.
1432 005242 012737 054076 000114      MOV      #SPUR,@#114
1433
1434      ;*****
1435      ;*TEST 1 PARITY ERROR ABORT
    
```

```

1436
1437
1438
1439
1440 005250 000004
1441 005252 012737 005424 001700
1442 005260 012737 000014 177746
1443 005266 012737 005334 001512
1444 005274 012737 005424 000114
1445 005302 012737 005360 000004
1446 005310 012737 005374 000014
1447 005316 012737 005410 000104
1448 005324 012704 170000
1449 005330 012702 177750
1450 005334 012706 001500
1451 005340 000401
1452
1453
1454
1455
1456 005344 000240
1457 005346 010412
1458 005350 005701
1459
1460 005352 005012
1461 005354 000240
1462 005356 104162
1463
1464 005360 005012
1465 005362 000240
1466 005364 012737 177777 177744
1467 005372 104163
1468 005374 005012
1469 005376 000240
1470 005400 012737 177777 177744
1471 005406 104164
1472 005410 005012
1473 005412 000240
1474 005414 012737 177777 177744
1475 005422 104165
1476
1477 005424 005012
1478 005426 000240
1479 005430 012737 177777 177744
1480 005436 012737 054050 000004
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491 005444 000004
    
```

```

: *
: * THIS TEST ENSURES THAT A CACHE PARITY ERROR FLAG CAUSES AN ABORT.
: * THIS IS DONE BY FORCING A PARITY ERROR ON AN EVEN WORD.
: *
: *****
TST1: SCOPE
      MOV #2$, $ESCAPE ; SETUP ESCAPE ADDRESS
      MOV #14, @CONTRL ; ENSURE MISSES TO BOTH GROUPS
      MOV #7$, $LPERR ; SETUP ERROR LOOP
      MOV #2$, @CACHVEC ; SETUP CACHE VECTOR
      MOV #3$, @ERRVEC ; SETUP LOCATION 4
      MOV #4$, @14 ; SETUP LOCATION 14
      MOV #5$, @104 ; SETUP LOCATION 104
      MOV #170000, R4 ; PUT MAINTENANCE DATA IN R4
      MOV @MAINT, R2 ; PUT ADDRESS OF MAIN REG IN R2
7$: MOV #STACK, SP ; INITIALIZE THE SP
      BR 1$ ; GO TO NEXT INSTRUCTION
      LOC= ; THIS IS
      LOC=-3@LOC ; USED TO MAKE
      LOC=LOC+4 ; 1$ FALL ON
      .=LOC ; AND EVEN WORD
1$: NOP ; USED TO MAKE BAD PARITY INSTR ON EVEN WORD
      MOV R4, (R2) ; SET BITS IN MAINT REG
      TST R1 ; EXECUTE INSTR TO CAUSE PE ABORT
: FAILURE, NO ABORT
      CLR (R2) ; CLEAR MAINT REG
      NOP
      ERROR 162 ; NO PE ABORT
: FAILURE, ABORTED TO WRONG VECTOR
3$: CLR (R2) ; ENSURE MAINT REG CLEAR
      NOP
      MOV #-1, @MEMERR
      ERROR 163 ; ABORTED TO LOCATION 4
4$: CLR (R2) ; ENSURE MAINT REG CLEAR
      NOP
      MOV #-1, @MEMERR
      ERROR 164 ; ABORTED TO 14
5$: CLR (R2) ; ENSURE MAINT REG CLEAR
      NOP
      MOV #-1, @MEMERR
      ERROR 165 ; ABORTED TO 104
: TEST OK
2$: CLR (R2) ; ENSURE MAINT REG CLEAR
      NOP
      MOV #-1, @MEMERR ; CLEAR MEMORY ERROR REG
      MOV #CPSPUR, @ERRVEC ; RESET LOCATION 4
      ; CONTINUE
: *****
: * 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 DOESN'T OCCUR THEN THE PROBLEM IS ON TMCA.
: * IF A TRAP OCCURS TO THE WRONG VECTOR THE PROBLEM COULD BE
: * ON TMCA OR UBCB.
: *
: *****
TST2: SCOPE
    
```

```

1492 005446 012737 005552 001700      MOV      #3$, $ESCAPE      ;SETUP ESCAPE ADDRESS
1493 005454 012737 005506 001512      MOV      #1$, $LPERR      ;SETUP ERROR LOOP
1494 005462 012737 005536 000004      MOV      #2$, @ERRVEC     ;SETUP THE ERROR VECTOR
1495 005470 012737 005552 000114      MOV      #3$, @CACHVEC    ;SETUP THE CACHE VECTOR
1496 005476 012704 170000      MOV      #170000, R4      ;PUT MAINT DATA IN R4
1497 005502 012702 177750      MOV      @MAINT, R2       ;PUT MAINT REG ADDR IN R2
1498 005506 012706 001500      MOV      #STACK, SP      ;INITIALIZE THE SP
1499 005512 000402      BR       4$              ;GO TO NEXT INSTRUCTION
1500      005514      LOC=      ;THIS IS USED
1501      005514      LOC=-3&LOC ;TO MAKE
1502      005520      LOC=LOC+4 ;1$ FALL ON
1503      005520      .=LOC      ;AN EVEN WORD
1504 005520 000240      4$:      NOP              ;GOOD PARITY ON EVEN WORD
1505 005522 010412      MOV      R4, (R2)        ;SET BITS IN MAINT REG
1506 005524 000240      NOP
1507 005526 005701      TST      R1
1508      ;FAILURE, NO TRAP
1509 005530 005012      CLR      (R2)           ;ENSURE MAINT REG CLEAR
1510 005532 000240      NOP
1511 005534 104166      ERROR   166            ;NO PE TRAP
1512      ;FAILURE, TRAPPED TO WRONG VECTOR
1513 005536 005012      2$:      CLR      (R2)           ;ENSURE MAINT REG CLEAR
1514 005540 000240      NOP
1515 005542 012737 177777 177744      MOV      #-1, @MEMERR    ;CLEAR MEM ERROR REG
1516 005550 104167      ERROR   167            ;PE TRAP, TRAPPED TO
1517      ;TEST OK
1518 005552 005012      3$:      CLR      (R2)           ;ENSURE MAINT REG CLEAR
1519 005554 000240      NOP
1520 005556 012737 177777 177744      MOV      #-1, @MEMERR    ;CLEAR MEM ERROR REG
1521 005564 012737 054050 000004      MOV      #CPSPUR, @ERRVEC ;RESTORE LOCATION 4
1522 005572 012737 054076 000250      MOV      #SPUR, @MMVEC   ;RESTORE MEM VEC
1523      ;CONTINUE
1524      ;*****
1525      ;*TEST 3      MEM MGT AND PE TRAP PRIORITY ARBITRATION
1526      ;*
1527      ;*      THIS TEST ENSURES THAT THE ARBITRATION LOGIC WORKS FOR MEMORY
1528      ;*      MANAGEMENT AND PARITY ERROR TRAPS.
1529      ;*
1530      ;*****
1531 005600 000004      TST3:   SCOPE
1532 005602 012737 001400 172354      1$:      MOV      #1400, @KIPAR6 ;RESTORE PAR6
1533 005610 112737 000004 172314      MOV      #4, @KIPDR6     ;SETUP PAGE 6 TO TRAP ON ALL ACCESSES
1534 005616 012704 170000      MOV      #170000, R4     ;PUT MAINT REG DATA IN R4
1535 005622 012702 177750      MOV      @MAINT, R2      ;PUT ADDRESS OF MAINT REG IN R2
1536      ;*****
1537      ;PIR6 DISABLED BY MGMT
    
```



```
1538 005626 012737 040000 140000 MOV #BIT14,@#140000 ;PUT PIR6 ENABLE BIT IN PAGE 6
1539 005634 012737 005706 000240 MOV #3$,@#PIRQVEC ;SETUP PIRQ VECTOR
1540 005642 012737 000340 000252 MOV #PR7,@#MMVEC+2 ;SET UP MMVEC PSW
1541 005650 012737 005720 000250 MOV #4$,@#MMVEC ;SETUP MEM MGMT VECTOR
1542 005656 012737 005664 001512 MOV #5$,@#SLPERR ;SETUP ERROR LOOP
1543 005664 012706 001500 5$: MOV #STACK,SP ;INITIALIZE THE SP
1544 005670 012737 001001 177572 MOV #1001,@#MMRO ;TURN RELOCATION ON
1545 005676 000235 SPL 5 ;SET PROCESSOR AT LEVEL 5
1546 005700 013737 140000 177772 MOV @#140000,@#PIRQ ;SET PIR6 AND MEM MGT TRAP
1547 :FAILURE, PIR6 CAME THRU
1548 005706 005037 177572 3$: CLR @#MMRO ;TURN RELOCATION OFF
1549 005712 005037 177772 CLR @#PIRQ ;CLEAR PIR6
1550 005716 104170 ERROR 170 ;PIR6 CAME IN ON
1551 :
1552 :*****
1553 :PIR3 DISABLED BY MGMT
1554 005720 005037 177572 4$: CLR @#MMRO ;TURN RELOCATION OFF
1555 005724 005037 177772 CLR @#PIRQ ;CLEAR PIR LEVEL 6
1556 005730 012737 006002 000240 MOV #6$,@#PIRQVEC ;SETUP PIRQ VECTOR
1557 005736 012737 006014 000250 MOV #7$,@#MMVEC ;SETUP MEM MGMT VECTOR
1558 005744 012737 005760 001512 MOV #8$,@#SLPERR ;SETUP ERROR LOOP
1559 005752 012737 004000 140000 MOV #BIT11,@#140000 ;PUT PIR3 ENABLE BIT IN PAGE 6
1560 005760 012706 001500 8$: MOV #STACK,SP ;INITIALIZE THE SP
1561 005764 012737 001001 177572 MOV #1001,@#MMRO ;TURN ON RELOCATION
1562 005772 000232 SPL 2 ;LOWER CPU TO LEVEL 2
1563 005774 013737 140000 177772 MOV @#140000,@#PIRQ ;SET PIR3 & MGMT
1564 :FAILURE, PIR3 CAME THRU
1565 006002 005037 177572 6$: CLR @#MMRO ;TURN OFF RELOCATION
1566 006006 005037 177772 CLR @#PIRQ ;CLEAR PIR3
1567 006012 104171 ERROR 171 ;PIR3 CAME IN ON
1568 :
1569 :*****
1570 :STACK LIMIT YELLOW DISABLED BY PARITY ERROR
1571 006014 005037 177572 7$: CLR @#MMRO ;TURN RELOCATION OFF
1572 006020 005037 177772 CLR @#PIRQ ;CLEAR PIR LEVEL 3
1573 006024 012737 006064 001512 MOV #9$,@#SLPERR ;SETUP ERROR LOOP
1574 006032 012737 006110 000004 MOV #10$,@#ERRVEC ;SETUP THE ERROR VECTOR
1575 006040 012737 006110 000114 MOV #10$,@#CACHVEC ;SETUP CACHEVEC
1576 006046 012737 000240 000116 MOV #PR5,@#CACHVEC+2 ;PUT PRIORITY 5 IN CACHE VECTOR PSW
1577 006054 012704 170000 MOV #170000,R4 ;PUT MAINT REG DATA IN R4
1578 006060 012702 177750 MOV #MAINT,R2 ;PUT ADDRESS OF MAINT ON R2
1579 006064 005037 000370 9$: CLR @#370 ;ENSURE LOCATION 370 CLEAR
1580 006070 012706 000376 MOV #376,SP ;SETUP THE SP TO YELLOW ZONE
1581 006074 000401 BR 11$ ;GO TO 12$
1582 006076 LOC= ;THIS MAKES
1583 006074 LOC=-3&LOC ;THE NEXT INSTRUCTION
1584 006100 LOC=LOC+4 ;FALL ON
1585 006100 .=LOC ;AN EVEN WORD
1586 006100 010412 11$: MOV R4,(R2) ;SET MAINT REG
1587 006102 000240 NOP ;ODD WORD GOOD PARITY
1588 006104 005216 INC (SP) ;CAUSE YEL ZONE (GOOD PARITY)
1589 006106 005701 TST R1 ;ODD WORD BAD PARITY
1590 :SHOULD TAKE PE TRAP THEN YEL ZONE TRAP
1591 006110 005012 10$: CLR (R2) ;CLEAR MAINTENANCE REGISTER
1592 006112 000240 NOP
1593 006114 022737 000240 000370 CMP #PR5,@#370 ;DID CACHVEC PSW GET STACKER?
```

```
1594 006122 001403          BEQ      12$          ;BRANCH IF YES
1595 006124 012706 001500    MOV      #STACK,SP   ;RESTORE THE SP
1596 006130 104172          ERROR    172         ;YEL ZONE CAME THRU ON PE TRAP
1597                                     ;*****
1598                                     ;MEMORY MANAGEMENT TRAP DISABLED BY PARITY TRAP
1599 006132 012737 006200 001512 12$:  MOV      #13$,SLPERR ;SETUP ERROR LOOP
1600 006140 012737 006230 000250    MOV      #15$,@MMVEC ;SETUP MEM MGT VECTOR
1601 006146 012737 006230 000114    MOV      #15$,@CACHVEC ;SETUP CACHVEC
1602 006154 012737 000340 000116    MOV      #PR7,@CACHVEC+2 ;RESTORE EACH VEC PSW
1603 0061 2 012704 170000          MOV      #170000,R4   ;PUT MAINT DATA IN R4
1604 006166 012702 177750          MOV      #MAINT,R2    ;PUT ADDRESS OF MAINT REG IN R2
1605 006172 112737 000004 172314    MOV      #4,@KIPDR6   ;ENSURE PAGE 6 TRAPS
1606 006200 012706 001500 13$:  MOV      #STACK,SP   ;INITIALIZE THE SP
1607 006204 012737 001001 177572    MOV      #1001,@MMRO  ;TURN RELOCATION ON
1608 006212 000402          BR       16$
1609          006214          LOC=.
1610          006214          LOC=-3&LOC
1611          006220          LOC=LOC+4
1612          006220          .=LOC
1613 006220 010412 16$:  MOV      R4,(R2)      ;SET MAINT REG (PARITY GOOD)
1614 006222 000240          NOP
1615 006224 005237 140402          INC      @#140402    ;ODD WORD PARITY GOOD
1616                                     ;INC HAS GOOD PARITY BUT ADDRESS
1617                                     ;HAS BAD PARITY. CAUSES MM TRAP
1618                                     ;AND PE TRAP
1619 006230 005012 15$:  CLR      (R2)        ;CLEAR MAINT REG
1620 006232 000240          NOP
1621 006234 005037 177572          CLR      @MMRO       ;TURN RELOCATION OFF
1622 006240 026627 000002 000340    CMP      2(SP),#PR7   ;DID PE TRAP OCCUR FIRST?
1623 006246 001401          BEQ      14$         ;BRANCH IF YES
1624 006250 104173          ERROR    173         ;MEM MGT TRAP CAME
1625 006252 012737 054050 000004 14$:  MOV      #CPSPUR,@ERRVEC ;RESTORE LOCATION 4
1626 006260 012737 054076 000114    MOV      #SPUR,@CACHVEC ;RESTORE LOCATION 114
1627 006266 012737 177777 177744    MOV      #-1,@MEMERR  ;CLEAR MEM ERROR REG
1628 006274 005037 177766          CLR      @CPUERR     ;ENSURE CPUERROR CLEAR
1629                                     ;CONTINUE
1630                                     ;
1631                                     ;*****
1632                                     ;* THE NEXT TEST USES THE MAPPING BOX AND THE CACHE TO
1633                                     ;* GENERATE A PARITY ERROR ON THE UNIBUS.
1634                                     ;*****
1635                                     ;*TEST 4 UNIBUS PARITY ERROR
1636                                     ;*
1637                                     ;* THIS TEST MAKES A REFERENCE TO MEMORY THRU THE MAPPING
1638                                     ;* BOX THAT WILL CAUSE A PARITY ERROR. IF THE ABORT DOESN'T
1639                                     ;* HAPPEN THEN THE PROBLEM IS ON UBCB.
1640                                     ;*
1641                                     ;* NOTE: MAP REGISTER 0 AND 1 ARE NOT USED IN CASE THE PROGRAM
1642                                     ;* IS RUNNING UNDER ACT11.
1643                                     ;*****
1644 006300 000004 TST4:  SCOPE
1645 006302 012737 077406 172314    MOV      #77406,@KIPDR6 ;SETUP PDR6
1646 006310 012737 000060 172516    MOV      #60,@MMR3    ;SETUP MMR3
1647 006316 012706 001500          MOV      #STACK,SP   ;INITIALIZE THE SP
1648 006322 012700 170210          MOV      #MAPL2,R0    ;GET ADDRESS OF MAP REG 2
1649 006326 012701 000036          MOV      #36,R1      ;SETUP SOB COUNT
```

```

1650 006332 012737 006344 000004      MOV    #55,@#ERRVEC    ;SETUP ERROR VECTOR
1651 006340 005720                    8$:   TST    (R0)+        ;SEE IF MAP REG IS ENABLED
1652 006342 000420                    BR     6$              ;BRANCH IF YES
1653 006344 062700 000002            5$:   ADD    #2,R0          ;ADJUST R0 TO NEXT REGISTER
1654 006350 077105                    SOB    R1,8$          ;TEST NEXT REGISTER
1655 006352 012706 001500            7$:   MOV    #STACK,SP     ;RESTORE THE SP
1656 006356 005737 001500            TST    $PASS         ;FIRST PASS?
1657 006362 001105                    BNE    $EOT           ;BRANCH IF NO
1658 006364 032737 040000 177570     BIT    #SW14,@#SWR    ;IS TEST BEING LOOPED ON?
1659 006372 001101                    BNE    $EOT           ;BRANCH IF YES
1660 006374 104400 070326            TYPE   ,EM724         ;TYPE MESSAGE
1661 006400 000137 006576            JMP    $EOT           ;GO TO NEXT TEST
1662 006404 005010                    6$:   CLR    (R0)          ;ENSURE MAP REG HIGH CLEAR
1663 006406 162700 000002            SUB    #2,R0          ;GET ADDR OF MAP REG LOW
1664 006412 012710 140000            MOV    #140000,(R0)   ;PUT ADDR OF PAGE 6 IN MAP REG
1665 006416 072027 000005            ASH    #5,R0          ;ADJUST ADDR FOR PAR6
1666 006422 052700 170000            BIS    #170000,R0    ;SET UNIBUS ADDR BITS
1667 006426 010037 172354            MOV    R0,@#KIPAR6   ;PUT IN PAGE 6 PAR
1668 006432 012737 005701 140000     MOV    #5701,@#140000 ;PUT WORD WITH PAD PARITY IN 140000
1669 006440 012704 170000            MOV    #170000,R4    ;PUT MAINT REG DATA IN R4
1670 006444 012702 177750            MOV    #MAINT,R2     ;PUT ADDRESS OF MAINT REG IN R2
1671 006450 012737 006472 001512     MOV    #1$,$LPERR    ;SETUP ERROR LOOP
1672 006456 012737 006534 000000     MOV    #4$,@#0       ;SETUP LOCATION ZERO
1673 006464 012737 006554 000114     MOV    #2$,@#CACHVEC ;SETUP CACH VECTOR
1674 006472 012706 001500            1$:   MOV    #STACK,SP     ;INITIALIZE THE SP
1675 006476 052737 000001 177572     BIS    #BIT0,@#MMR0  ;TURN RELOCATION ON
1676 006504 000401                    BR     3$              ;GO TO TEST
1677                                LOC=
1678                                !OC--38! OC
1679                                LOC=LOC+4
1680                                --LOC
1681 006510 010412                    3$:   MOV    R4,(R2)     ;SET BITS IN MAINT REG
1682 006512 000240                    NOP                                ;GOOD PARITY ON ODD WORD
1683 006514 005037 140000            CLR    @#140000      ;EXECUTE A DATIP THRU THE
1684                                ;MAP THAT CAUSES A PE
1685                                ;FAILURE, NO ABORT
1686 006520 005012                    CLR    (R2)          ;CLEAR MAINT REG
1687 006522 000240                    NOP                                ;
1688 006524 005037 177572            CLR    @#MMR0        ;TURN RELOCATION OFF
1689 006530 104174                    ERROR  174           ;NO UNIBUS PE ABORT
1690 006532 000410                    BR     2$              ;
1691                                ;TRAPPED TO WRONG VECTOR
1692 006534 005012                    4$:   CLR    (R2)     ;ENSURE MAINT REG CLEAR
1693 006536 000240                    NOP                                ;
1694 006540 005037 177572            CLR    @#MMR0        ;TURN OFF RELOCATION
1695 006544 012737 177777 000004     MOV    #-1,@#ERRVEC  ;CLEAR ERROR REGISTER
1696 006552 104175                    ERROR  175           ;TRAPPED TO ZERO
1697                                ;TEST OK
1698 006554 005012                    2$:   CLR    (R2)     ;ENSURE MAINT REG CLEAR
1699 006556 000240                    NOP                                ;
1700 006560 005037 177572            CLR    @#MMR0        ;TURN RELOCATION OFF
1701 006564 012737 177777 177744     MOV    #-1,@#MEMERR  ;CLEAR ERROR REG
1702 006572 005037 172516            CLR    @#MMR3        ;ENSURE MAP TURNED OFF
1703                                $EOT:
1704                                ;*****
1705                                ;*TEST 5          CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ONES
    
```

```
1706 : *
1707 : *THIS TEST IS A TEST OF BOTH THE AMX, CPU INPUTS, AND
1708 : *THE CACHE ERROR ADDRESS REGISTER. A SET OF ADDRESSES IS
1709 : *GENERATED AND A MAIN MEMORY ADDRESS AND CONTROL LINE
1710 : *PARITY ERROR IS FORCED AT EACH, THEREBY LOCKING UP
1711 : *THE ADDRESS ON THE OUTPUT OF THE AMX IN THE ERROR
1712 : *ADDRESS REGISTER. THE MANNER IN WHICH THIS IS DONE
1713 : *IS AS FOLLOWS: FIRST THE ADDRESS IS GENERATED;
1714 : *THEN, IF IT IS A VALID ADDRESS (THAT IS, IF IT IS NOT
1715 : *BEYOND THE LIMITS OF MEMORY AS DISPLAYED IN THE
1716 : *SYSTEM SIZE REGISTER), THESE THREE INSTRUCTIONS ARE MOVED
1717 : *TO THAT AREA OF MEMORY:
1718 : *   ONE:  MOV   R1,(R2)
1719 : *   2$:  CLR   (R2)
1720 : *   3$:  RTS   PC
1721 : *2$ IS THE ADDRESS BEING TESTED. THE INSTRUCTION
1722 : *AT ONE IS GIVEN CONTROL BY A 'JSR PC'. R1 IS MADE
1723 : *TO CONTAIN #2 AND R2 CONTAINS THE ADDRESS OF
1724 : *THE MAINTENANCE REGISTER, SO THAT AFTER THE 'MOV R1,(R2)'
1725 : *IS EXECUTED A PARITY ERROR SHOULD OCCUR ON THE
1726 : *MAIN MEMORY ADDRESS AND CONTROL LINES WHEN THE
1727 : *NEXT INSTRUCTION IS FETCHED.
1728 : *THE ADDRESSES USED ARE GENERATED FOLLOWINT THIS PATTERN
1729 : *
1730 : *      200000
1731 : *      200002
1732 : *      200004
1733 : *      200010
1734 : *      200020
1735 : *      200040
1736 : *      200100
1737 : *      200200
1738 : *      200400
1739 : *      ETC. TO:
1740 : *      240000
1741 : *      300000
1742 : *      400000
1743 : *      400002
1744 : *      400004
1745 : *      400010
1746 : *      ETC. TO:
1747 : *      500000
1748 : *      600000
1749 : *      1000000
1750 : *      1000002
1751 : *      1000004
1752 : *      ETC.
1753 : *THE PATTERN CONINUES UNTIL AN ADDRESS IS GENERATED THAT
1754 : *IS TOO LARGE.
1755 : *MEMORY MANAGEMENT IS SET UP TO FULL 22-BIT MODE, SO
1756 : *IF THE USER WANTS TO HAVE THE EXECUTION OF THIS
1757 : *TEST DELETED HE CAN SIMPLY BY TURNING ON THE APPORPRIATE
1758 : *CONSOLE SWITCH WHICH HAS BEEN DESIGNATED FOR THE
1759 : *PURPOSE OF DELETING THE EXECUTION OF TESTS WHICH
1760 : *MAKE USER OF MEMORY MANAGEMENT.
1761 : *
```

```

1762 006576 000004 TST5: SCOPE
1763 006600 012737 000020 001676 MOV #20,$TIMES ;;DO 20 ITERATIONS
1764 000005 X-$TN-1
1765
1766 006606 012737 007504 054230 MOV #TST6,SKAD ;SET THE SKAD REGISTER
1767 ;IN CASE THE TEST ABORTS.
1768 006614 113737 001502 001626 MOVB $TSTM,$TMP0
1769 006622 012737 054076 000114 MOV #SPUR,@CACHVEC ;INITIALLY EXPECT NO ERRORS
1770
1771 ;SEE IF THIS TEST SHOULD
1772 ;BE EXECUTED. THE CONDITION
1773 ;TEST IS THE DESIGNATED
1774 006630 104422 MMSKIP ;CONSOLE SWITCH.
1775 006632 012700 172340 MOV #KIPAR0,R0 ;INITIALIZE THE KERNAL
1776 006636 012701 077406 MOV #77406,R1 ;SPACE MEMORY MANAGEMENT
1777 006642 012702 172300 MOV #KIPDR0,R2 ;REGISTERS
1778 006646 012703 000010 MOV #10,R3
1779 006652 010122 1$: MOV R1,(R2)+
1780 006654 077302 SOB R3,1$
1781 006656 005020 CLR (R0)+
1782 006660 012720 000200 MOV #200,(R0)+
1783 006664 012720 000400 MOV #400,(R0)+
1784 006670 012720 000600 MOV #600,(R0)+
1785 006674 012720 001000 MOV #1000,(R0)+
1786 006700 012720 001200 MOV #1200,(R0)+
1787 006704 012720 001400 MOV #1400,(R0)+
1788 006710 012710 177600 MOV #177600,(R0)
1789 006714 012737 000020 172516 MOV #20,@MMR3 ;TURN ON MEMORY MANAGEMENT
1790 006722 012737 000001 177572 MOV #1,@MMR0
1791 006730 104424 SIZE ;DETERMINE FROM THE SYSTEM
1792 ;SIZE REGISTER WHAT THE
1793 ;HIGHEST ADDRESSABLE WORD
1794 ;OF MEMORY IS.
1795 006732 000000 XLOADR: .WORD 0 ;LOW ORDER 16-BITS OF THE
1796 006734 000000 XHIADR: .WORD 0 ;ADDRESS AND HIGH ORDER 6-BITS
1797 006736 042737 000002 006732 BIC #2,XLOADR ;SET THE HIGHEST WORD MINUS TWO
1798 ;IN XLOADR.
1799
1800 006744 012737 000014 177746 MOV #MOM1,@CONTRL ;FORCE MISSES TO BOTH GROUPS.
1801
1802 006752 005037 007472 CLR XADR3 ;INITIALIZE STORAGE
1803 006756 005037 007474 CLR XADR3+2 ;LOCATIONS USED TO GENERATE
1804 006762 005037 007462 CLR XADR1 ;THE SERIES OF TEST ADDRESSES.
1805 006766 012737 000001 007464 MOV #1,XADR1+2
1806
1807 006774 X1:
1808 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
1809
1810 006774 023737 007464 007474 CMP XADR1+2,XADR3+2 ;COMPARE THE HIGH ORDER
1811 007002 001006 BNE 64$ ;PARTS OF XADR1 AND ARG2.
1812 007004 023737 007462 007472 CMP XADR1,XADR3 ;COMPARE THE LOW ORDER
1813
1814 007012 001002 BNE 64$ ;PARTS.
1815
1816
1817
    
```

J 5

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 36
 CEKBD.P11 16-MAY-79 08:58 T5 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ONES SEQ 0061

1818	007014	000137	007436		JMP	X11		;THEY WERE EQUAL!
1819								
1820	007020	103402		64\$:	BLO	65\$		
1821	007022	000137	007032		JMP	X2		;THE FIRST ADDRESS IS LARGER
1822								;THAN THE SECOND.
1823	007026	000137	007436	65\$:	JMP	X11		;THE FIRST IS LESS THAN THE
1824								;SECOND.
1825								
1826								

```

1827 007032
1828
1829 007032 013737 007462 007466
1830 007040 013737 007464 007470
1831 007046 063737 007472 007466
1832 007054 005537 007470
1833 007060 063737 007474 007470
1834
1835
1836
1837
1838 007066
1839
1840
1841 007066 023737 007470 006734
1842 007074 001006
1843 007076 023737 007466 006732
1844
1845 007104 001002
1846
1847
1848
1849 007106 000137 007502
1850
1851 007112 103402
1852 007114 000137 007502
1853
1854 007120 000137 007124
1855
1856
1857 007124 012737 007124 001512
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875 007132 013703 007466
1876 007136 013702 007470
1877 007142 162703 000002
1878 007146 005602
1879
1880 007150 010300
1881 007152 042700 177701
1882 007156 062700 140000
    
```

```

X2:
;DOUBLE PRECISION ADDITION, UNSIGNED
MOV XADR1,XADR2
MOV XADR1+2,XADR2+2
ADD XADR3,XADR2
ADC XADR2+2
ADD XADR3+2,XADR2+2

X3:
;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
CMP XADR2+2,XLOADR+2 ;COMPARE THE HIGH ORDER
BNE 64$ ;PARTS OF XADR2 AND ARG2.
CMP XADR2,XLOADR ;COMPARE THE LOW ORDER
BNE 64$ ;PARTS.
JMP XDONE ;THEY WERE EQUAL.
64$: BLO 65$
JMP XDONE ;THE FIRST ADDRESS IS LARGER
;THAN THE SECOND!
65$: JMP X4 ;THE FIRST IS LESS THAN THE
;SECOND.
X4: MOV #X4,$LPERR
;CONVERT THE 22-BIT ADDRESS IN XADR2 TO VIRTUAL ADDRESS
;WHICH WILL RELOCATE THROUGH KIPAR6; SET UP KIPAR6;
;TURN ON MEMORY MANAGEMENT; PUT THE INSTRUCTIONS:
: 1$: MOV R1,(R2)
: 2$: CLR (R2)
: 3$: RTS PC
;AT THE LOCATION BEING TESTED, WITH 2$=TEST ADDRESS;
;PUT A PATTERN,000002, IN R1 FOR THE MAINTENANCE
;REGISTER TO FORCE BAD PARITY ON THE MAIN MEMORY
;ADDRESS AND CONTROL LINES. PUT THE ADDRESS OF
;THE CACHE MAINTENANCE REGISTER IN R2. PUT THE
;ADDRESS, X6, IN LOCATION CACHVEC TO TAKE CARE OF THE
;WHICH IS BEING FORCED. JSR TO THE ABOVE ROUTINE,
;SO THAT IF THE PARITY ERROR DOES'NT OCCUR
;THE 'RTS PC', AT 3$ ABOVE, WILL HANDLE IT.
MOV XADR2,R3
MOV XADR2+2,R2
SUB #2,R3
SBC R2
MOV R3,R0
BIC #177701,R0
ADD #140000,R0
    
```

```

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) L 5 16-MAY-79 09:11 PAGE 38
CEKBDD.P11 16-MAY-79 08:58 T5 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ONES SEQ 0063

1883 007162 073227 177772 ASHC #-6,R2
1884 007166 010337 172354 MOV R3,@#KIPAR6
1885
1886 007172 012737 000020 172516 MOV #20,@MMR3 ;TURN ON MEMORY
1887 007200 012737 000001 177572 MOV #1,@MMR0 ;MANAGEMENT.
1888 ;SET UP THE TEST INSTRUCTIONS.
1889 007206 012710 010112 MOV #010112,(R0) ;010112 = 'MOV R1,(R2)'
1890 007212 012760 005012 000002 MOV #005012,2(R0) ;005012 = 'CLR (R2)'
1891 007220 012760 000207 000004 MOV #000207,4(R0) ;000207 = 'RTS PC'
1892
1893 007226 012701 000002 MOV #2,R1 ;SET UP THE REGISTERS
1894 007232 012702 177750 MOV #MAINT,R2
1895
1896 007236 012737 007256 000114 MOV #X6,@#CACHVEC ;SET UP THE PARITY ERROR
1897 007244 000240 NOP ;TRAP VECTOR AND GO.
1898 007246 004710 JSR PC,(R0)
1899
1900 007250 X5: ;NO TRAP OR ABORT OCCURRED!
1901 ;MAINTENANCE FUNCTION
1902 ;FOR BAD PARITY ON
1903 007250 104022 1$: ERROR 22 ;THE MAIN MEMORY ADDRESS
1904 007252 000137 007370 JMP X9 ;AND CONTROL LINES FAILED
1905
1906 007256 X6:
1907 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
1908 ;COMPARE THE HIGH ORDER
1909 007256 023737 007470 177742 CMP XADR2+2,LOADRS+2 ;PARTS OF XADR2 AND ARG2.
1910 007264 001006 BNE 64$ ;COMPARE THE LOW ORDER
1911 007266 023737 007466 177740 CMP XADR2,LOADRS ;PARTS.
1912
1913 007274 001002 BNE 64$ ;THEY WERE EQUAL!
1914
1915
1916
1917 007276 000137 007314 JMP X7 ;THE FIRST ADDRESS IS LARGER
1918 64$: BLO 65$ ;THAN THE SECOND!
1919 007302 103402 JMP X8 ;THE FIRST IS LESS THAN THE
1920 007304 000137 007332 65$: JMP X8 ;SECOND.
1921
1922 007310 000137 007332
1923
1924
1925 ;PARITY ERROR OCCURS.
1926 007314 005726 X7: TST (SP)+ ;RESTORE THE STACK.
1927 007316 022626 CMP (SP)+,(SP)+ ;AND CONTINUE SINCE
1928 007320 012737 177777 177744 MOV #-1,@MEMERR ;THE CACHE ERROR ADDRESS
1929 007326 000137 007370 JMP X9 ;REGISTER WAS SET CORRECTLY.
1930
1931 007332 013737 177744 001630 X8: MOV @MEMERR,$TMP1 ;REPORT VALID TEST
1932 ;FAILURE.
1933 007340 013737 177740 001634 MOV @LOADRS,$TMP3
1934 007346 013737 177742 001636 MOV @HIADRS,$TMP4
1935 007354 005726 TST (SP)+
1936 007356 022626 CMP (SP)+,(SP)+
1937 007360 104023 ERROR 23
1938 007362 012737 177777 177744 MOV #-1,@MEMERR

```



```
1939  
1940 007370 005037 177572          x9:  CLR @MMR0          ;TURN OFF MEMORY MANAGEMENT.  
1941 007374 005037 172516          CLR @MMR3  
1942 007400 005737 007472          TST XADR3  
1943 007404 001007                   BNE X10          ;GET READY TO GENERATE  
1944 007406 005737 007474          TST XADR3+2     ;THE NEXT TEST ADDRESS.  
1945 007412 001004                   BNE X10  
1946 007414 012737 000002 007472  MOV #2,XADR3  
1947 007422 000415                   BR X12
```

1948 007424 006337 007472
 1949 007430 006137 007474
 1950 007434 000410
 1951
 1952 007436 006337 007462
 1953 007442 006137 007464
 1954 007446 005037 007472
 1955 007452 005037 007474
 1956 007456 000137 006774
 1957
 1958 007462 000000
 1959 007464 000000
 1960 007466 000000
 1961 007470 000000
 1962 007472 000000
 1963 007474 000000
 1964 007476 000000
 1965 007500 000000
 1966 007502 104416

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
 XADR2: .WORD 0
 XADR3: .WORD 0
 XADR4: .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 WILL BE FLOATED THROUGH
 : *A FIELD OF ONES TO PRODUCE THE ADDRESSES
 : *BASE ADDRESSES WHICH ARE USED ARE:

: * 177776
 : * 377776
 : * 777776
 : * 1777776
 : * 3777776
 : * 7777776
 : * 17777776

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

1989
 1990 007504 000004
 1991 007506 012737 000020 001676
 1992 000006
 1993
 1994 007514 012737 010404 054230
 1995
 1996 007522 113737 001502 001626
 1997 007530 012737 054076 000114
 1998
 1999 007536 104422
 2000
 2001
 2002
 2003

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
 ;:SWITCH DESIGNATED AS
 ;:THE DON'T USE MEMORY

```
2004
2005 007540 012700 172340      MOV      #KIPAR0,R0      ;MANAGEMENT SWITCH.
2006 007544 012701 077406      MOV      #77406,R1      ;INITIALIZE THE KERNAL MODE
2007 007550 012702 172300      MOV      #KIPDR0,R2     ;MEMORY MANAGEMENT REGISTERS.
2008 007554 012703 000010      MOV      #10,R3
2009 007560 010122              1$:  MOV      R1,(R2)+
2010 007562 077302              SOB      R3,1$
2011 007564 005020              CLR      (R0)+
2012 007566 012720 000200      MOV      #200,(R0)+
2013 007572 012720 000400      MOV      #400,(R0)+
2014 007576 012720 000600      MOV      #600,(R0)+
2015 007602 012720 001000      MOV      #1000,(R0)+
2016 007606 012720 001200      MOV      #1200,(R0)+
2017 007612 012720 001400      MOV      #1400,(R0)+
2018 007616 012710 177600      MOV      #177600,(R0)
2019 007622 012737 000020 172516  MOV      #20,@MMR3      ;TRUN ON MEMORY MANAGEMENT
2020 007630 012737 000001 177572  MOV      #1,@MMR0
2021 007636 104424              SIZE
2022 007640 000000              XXL0A: .WORD 0      ;GET THE LARGEST MEMORY
2023 007642 000000              XXH1A: .WORD 0      ;WORD ADDRESS INTO XXL0A
2024 007644 042737 000002 007640  BIC      #2,XXL0A      ;AND XXH1A.
2025                                ;GET THE ADDRESS OF THE HIGHEST WORD
                                ;WORD MINUS TWO.
```

2026

D 6

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 43
 CEKBDD.P11 16-MAY-79 08:58 T6 CACHE ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FLOATING ZEROES SEQ 0068

```

2027 007652 012737 000014 177746      MOV      #MOM1, @#CONTRL      ;FROM NOW ON FORCE MISSES
2028                                     ;TO BOTH GROUPS.
2029
2030 007660 012737 177776 010362  XX1:   MOV      #177776, XXADR1      ;INITIALIZE
2031 007666 005037 010364                CLR      XXADR1+2
2032 007672 012704 000016                MOV      #16, R4
2033 007676 000410                BR       XX3
2034
2035 007700 005204                XX2:   INC      R4              ;TURN ON THE NEXT BIT
2036 007702 052737 000001 010362        BIS      #1, XXADR1          ;IN THE FIELD OF ONES.
2037 007710 006337 010362                ASL      XXADR1
2038 007714 006137 010364                ROL      XXADR1+2
2039
2040 007720 012737 000002 010372  XX3:   MOV      #2, XXMASK          ;INITIALIZE THE MASK
2041 007726 005037 010374                CLR      XXMASK+2           ;USED TO CREATE THE ZERO
2042                                     ;IN THE FIELD OF ONES.
2043 007732 010405                MOV      R4, R5
2044 007734 012737 007742 001512        MOV      #XX4, $LPERR
2045
2046 007742 013737 010362 010366  XX4:   MOV      XXADR1, XXADR2      ;DETERMINE THIS TEST ADDRESS.
2047 007750 013737 010364 010370        MOV      XXADR1+2, XXADR2+2
2048 007756 043737 010372 010366        BIC      XXMASK, XXADR2
2049 007764 043737 010374 010370        BIC      XXMASK+2, XXADR2+2
2050
2051
2052                                     ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2053 007772 023737 010370 010400        CMP      XXADR2+2, XXCNST+2  ;COMPARE THE HIGH ORDER
2054 010000 001006                BNE      64$                ;PARTS OF XXADR2 AND ARG2.
2055 010002 023737 010366 010376        CMP      XXADR2, XXCNST     ;COMPARE THE LOW ORDER
2056
2057 010010 001002                BNE      64$                ;PARTS.
2058
2059
2060
2061 010012 000137 010030                JMP      XX5                ;THEY WERE EQUAL.
2062
2063 010016 103402                64$:   BLO      65$
2064 010020 000137 010030                JMP      XX5                ;THE FIRST ADDRESS IS LARGER
2065                                     ;THAN THE SECOND.
2066 010024 000137 010320                65$:   JMP      XX10             ;THE FIRST IS LESS THAN THE
2067                                     ;SECOND.
2068
2069
2070 010030                XX5:
2071
2072                                     ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2073 010030 023737 010370 007642        CMP      XXADR2+2, XXLOA+2  ;COMPARE THE HIGH ORDER
2074 010036 001006                BNE      64$                ;PARTS OF XXADR2 AND ARG2.
2075 010040 023737 010366 007640        CMP      XXADR2, XXLOA     ;COMPARE THE LOW ORDER
2076
2077 010046 001002                BNE      64$                ;PARTS.
2078
2079
2080
2081 010050 000137 010066                JMP      XX6                ;THEY WERE EQUAL.
2082

```

```

2083 010054 103402          64$: BLO 65$
2084 010056 000137 010320      JMP  XX10      ;THE FIRST ADDRESS IS LARGER
2085                                     ;THAN THE SECOND!
2086 010062 000137 010066      65$: JMP  XX6      ;THE FIRST IS LESS THAN THE
2087                                     ;SECOND.
2088
2089
2090 010066

```

```

2091
2092                                     ;CONVERT THE 22-BIT ADDRESS IN XXADR2 TO VIRTUAL ADDRESS
2093                                     ;WHICH WILL RELOCATE THROUGH KIPAR6; SET UP KIPAR6;
2094                                     ;TURN ON MEMORY MANAGEMENT; PUT THE INSTRUCTIONS:
2095                                     :      1$:  MOV   R1,(R2)
2096                                     :      2$:  CLR   (R2)
2097                                     :      3$:  RTS   PC
2098                                     ;AT THE LOCATION BEING TESTED, WITH 2$=TEST ADDRESS;
2099                                     ;PUT A PATTERN,000002, IN R1 FOR THE MAINTENANCE
2100                                     ;REGISTER TO FORCE BAD PARITY ON THE MAIN MEMORY
2101                                     ;ADDRESS AND CONTROL LINES. PUT THE ADDRESS OF
2102                                     ;THE CACHE MAINTENANCE REGISTER IN R2. PUT THE
2103                                     ;ADDRESS, XX7, IN LOCATION CACHVEC TO TAKE CARE OF THE
2104                                     ;WHICH IS BEING FORCED. JSR TO THE ABOVE ROUTINE,
2105                                     ;SO THAT IF THE PARITY ERROR DOES'NT OCCUR
2106                                     ;THE 'RTS PC', AT 3$ ABOVE, WILL HANDLE IT.
2107

```

```

2108 010066 013703 010366      MOV   XXADR2,R3
2109 010072 013702 010370      MOV   XXADR2+2,R2
2110 010076 162703 000002      SUB   #2,R3
2111 010102 005602              SBC   R2
2112
2113 010104 010300              MOV   R3,R0
2114 010106 042700 177701      BIC   #177701,R0
2115 010112 062700 140000      ADD   #140000,R0
2116 010116 073227 177772      ASHC  #-6,R2
2117 010122 010337 172354      MOV   R3,@#KIPAR6
2118
2119 010126 012737 000020 172516  MOV   #20,@MMR3      ;TURN ON MEMORY
2120 010134 012737 000001 177572  MOV   #1,@MMR0      ;MANAGEMENT.
2121                                     ;SET UP THE TEST INSTRUCTIONS.
2122 010142 012710 010112      MOV   #010112,(R0)  ;010112 = 'MOV R1,(R2)'
2123 010146 012760 005012 000002  MOV   #005012,2(R0) ;005012 = 'CLR (R2)'
2124 010154 012760 000207 000004  MOV   #000207,4(R0) ;000207 = 'RTS PC'
2125
2126 010162 012701 000002      MOV   #2,R1          ;SET UP THE REGISTERS
2127 010166 012702 177750      MOV   #MAINT,R2
2128
2129 010172 012737 010210 000114  MOV   #XX7,@#CACHVEC ;SET UP THE PARITY ERROR
2130 010200 000240              NOP                    ;TRAP VECTOR AND GO.
2131 010202 004710              JSR   PC,(R0)
2132
2133                                     ;NO TRAP OCCURRED.

```

```

2134 010204 104024          1$:  ERROR 24
2135 010206 000444          BR   XX10
2136                                     ;COME HERE ON THE PARITY ERROR
2137 010210          XX7:
2138

```

```

2139 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2140 010210 023737 010370 177742 CMP XXADR2+2,LOADRS+2 ;COMPARE THE HIGH ORDER
2141 010216 001006 BNE 64$ ;PARTS OF XXADR2 AND ARG2.
2142 010220 023737 010366 177740 CMP XXADR2,LOADRS ;COMPARE THE LOW ORDER
2143
2144 010226 001002 BNE 64$ ;PARTS.
2145
2146
2147
2148 010230 000137 010246 JMP XX8 ;THEY WERE EQUAL.
2149
2150 010234 103402 64$: BLO 65$
2151 010236 000137 010262 JMP XX9 ;THE FIRST ADDRESS IS LARGER
2152 ;THAN THE SECOND
2153 010242 000137 010262 65$: JMP XX9 ;THE FIRST IS LESS THAN THE
2154 ;SECOND.
2155
2156
2157 010246 005726 XX8: TST (SP)+ ;RESTORE THE STACK.
2158 010250 022626 CMP (SP)+,(SP)+
2159 010252 012737 177777 177744 MOV #-1,@MEMERR ;RESET THE CACHE ERROR REGISTERS.
2160 010260 000417 BR XX10
2161 010262 013737 177744 001630 XX9: MOV @MEMERR,$TMP1 ;REPORT A VALID TEST
2162 ;FAILURE.
2163 010270 013737 177740 001634 MOV @LOADRS,$TMP3
2164 010276 013737 177742 001636 MOV @HIADRS,$TMP4
2165 010304 005726 TST (SP)+
2166 010306 022626 CMP (SP)+,(SP)+
2167 010310 104025 ERROR 25
2168 010312 012737 177777 177744 MOV #-1,@MEMERR
2169
2170 010320 006337 010372 XX10: ASL XXMASK ;ROTATE THE MASK.
2171 010324 006137 010374 ROL XXMASK+2
2172 010330 005305 DEC R5
2173 010332 001402 BEQ 1$
2174 010334 000137 007742 JMP XX4
2175 010340 005037 177572 1$: CLR @MMR0 ;TURN OF MEMORY MANAGEMENT.
2176 010344 005037 172516 CLR @MMR3
2177 010350 020427 000025 CMP R4,#25
2178 010354 002012 BGE XX11
2179 010356 000137 007700 JMP XX2
2180
2181 010362 000000 XXADR1: .WORD 0 ;USED TO GENERATE TEST PATTERNS.
2182 010364 000000 .WORD 0
2183 010366 000000 XXADR2: .WORD 0 ;USED TO STORE THE CURRENT
2184 010370 000000 .WORD 0 ;TEST PATTERN DURING A TEST.
2185 010372 000000 XXMASK: .WORD 0 ;MASK USED TO PUT A ZERO
2186 010374 000000 .WORD 0 ;IN THE FIELD OF ONES
2187 ;TO CREATE A TEST ADDRESS.
2188 010376 124704 XXCNST: .WORD BOTPRG ;THE SMALLEST ADDRESS
2189 010400 000000 .WORD 0 ;IN MEMORY OVER THIS TEST.
2190
2191 010402 104416 XX11: RSET
2192
2193 ;*****
2194 ;*TEST 7 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ONES
    
```

```

2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206 010404 000004
2207 010406 012737 000020 001676
2208 000007
2209
2210 010414 012737 011316 054230
2211
2212 010422 113737 001502 001626
2213 010430 012737 054076 000114
2214 010436 012737 054050 000004
2215
2216 010444 104422
2217
2218 010446 012700 172340
2219 010452 012701 077406
2220 010456 012702 172300
2221 010462 012703 000010
2222 010466 010122
2223 010470 077302
2224 010472 005020
2225 010474 012720 000200
2226 010500 012720 000400
2227 010504 012720 000600
2228 010510 012720 001000
2229 010514 012720 001200
2230 010520 012720 001400
2231 010524 012710 177600
2232
2233 010530 012737 000060 172516
2234 010536 012737 000001 177572
2235
2236 010544 104424
2237 010546 000000
2238 010550 000000
2239
2240 010552 042737 000002 010546
2241
2242 010560 012737 000014 177746
2243
2244 010566 005037 011310
2245 010572 005037 011312
2246 010576 005037 011300
2247 010602 012737 000001 011302
2248
2249 010610
2250

```

```

: *
: * THIS IS A TEST OF THE UNIBUS INPUTS TO THE AMX.
: * THIS TEST IS IDENTICAL TO TST5 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.
: *
: *****
TST7:  SCOPE
      MOV  #20,$TIMES      ;;DO 20 ITERATIONS
RR=$TN-1
      MOV  #TST10,SKAD     ;SET THE SKAD REGISTER
                          ;IN CASE THE TEST ABORTS.
      MOVB $TSTNM,$TMP0
      MOV  #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
      MOV  #CPSPUR,@#ERRVEC
      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)
      MOV  #60,@#MMR3     ;TURN ON MEMORY MANAGEMENT.
      MOV  #1,@#MMR0
      SIZE ;DETERMINE THE MEMORY
      RRLOAD: .WORD 0     ;SYSTEM SIZE.
      RRHIAD: .WORD 0     ;LOW ORDER 16-BITS AND
                          ;HIGH ORDER 6-BITS OF THE
                          ;HIGHEST MEMORY WORD ADDRESS.
      BIC  #?,RRLOAD      ;GET THE HIGHEST WORD IN MEMORY
                          ;MINUS TWO.
      MOV  #MOM1,@#CONTRL ;FORCE MISSES TO BOTH GROUPS
      CLR  RRADR3         ;INITIALIZE STORAGE LOCATIONS
      CLR  RRADR3+2       ;USED TO GENERATE THE
      CLR  RRADR1         ;SERIES OF TEST ADDRESSES.
      MOV  #1,RRADR1+2
RR1:

```



```

2251 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2252 010610 023737 011302 011312 CMP RRADR1+2,RRADR3+2 ;COMPARE THE HIGH ORDER
2253 010616 001006 BNE 64$ ;PARTS OF RRADR1 AND ARG2.
2254 010620 023737 011300 011310 CMP RRADR1,RRADR3 ;COMPARE THE LOW ORDER
2255
2256 010626 001002 BNE 64$ ;PARTS.
2257
2258
2259
2260 010630 000137 011254 JMP RR1' ;THEY WERE EQUAL!
2261
2262 010634 103402 64$: BLO 65$
2263 010636 000137 010646 JMP RR2 ;THE FIRST ADDRESS IS LARGER
2264 ;THAN THE SECOND!
2265 010642 000137 011254 65$: JMP RR11 ;THE FIRST IS LESS THAN THE
2266 ;SECOND.
2267
2268
2269 010646 RR2:
2270 ;DOUBLE PRECISION ADDITION, UNSIGNED
2271 010646 013737 011300 011304 MOV RRADR1,RRADR2
2272 010654 013737 011302 011306 MOV RRADR1+2,RRADR2+2
2273 010662 063737 011310 011304 ADD RRADR3,RRADR2
2274 010670 005537 011306 ADC RRADR2+2
2275 010674 063737 011312 011306 ADD RRADR3+2,RRADR2+2
2276
2277
2278
2279
2280 010702 RR3:
2281
2282 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2283 010702 023737 011306 010550 CMP RRADR2+2,RRLOAD+2 ;COMPARE THE HIGH ORDER
2284 010710 001006 BNE 64$ ;PARTS OF RRADR2 AND ARG2.
2285 010712 023737 011304 010546 CMP RRADR2,RRLOAD ;COMPARE THE LOW ORDER
2286
2287 010720 001002 BNE 64$ ;PARTS.
2288
2289
2290
2291 010722 000137 011314 JMP RRDONE ;THEY WERE EQUAL.
2292
2293 010726 103402 64$: BLO 65$
2294 010730 000137 011314 JMP RRDONE ;THE FIRST ADDRESS IS LARGER
2295 ;THAN THE SECOND.
2296 010734 000137 010740 65$: JMP RR4 ;THE FIRST IS LESS THAN THE
2297 ;SECOND.
2298
2299 010740 012737 010740 001512 RR4: MOV #RR4,$LPERR
2300 ;CONVERT THE PHYSICAL 22-BIT, ADDRESS IN RRADR2 TO A VIRTUAL ADDRESS
2301 ;WHICH WILL RELOCATE THROUGH KIPAR6 TO THE UNIBUS, THEN THROUGH
2302 ;THE MAPPING BOX TO THE UNIBUS INPUTS OF THE CACHE AMX.
2303 010746 013737 011304 170200 MOV RRADR2,@MAPL00 ;SET UP THE MAP REGISTER 0.
2304 010754 013737 011306 170202 MOV RRADR2+2,@MAPH00
2305 010762 162737 000002 170200 SUB #2,@MAPL00
2306 010770 005637 170202 SBC @MAPH00
    
```

```

2307
2308 010774 012700 140000      MOV      #140000,R0      ;A VIRTUAL ADDRESS WHICH WILL
2309                                ;RELOCATE THROUGH KIPAR6.
2310 011000 012737 170000 172354  MOV      #170000,@#KIPAR6;RELOCATE TO UNIBUS BASE
2311                                ;ADDRESS OF 000000.
2312 011006 012737 000060 172516  MOV      #60,@#MMR3     ;TURN ON THE MAPPING BOX AND
2313                                ;22-BIT MODE.
2314 011014 012737 000001 177572  MOV      #1,@#MMR0     ;TURN ON MEMORY MANAGEMENT.
2315                                ;SET UP THE TEST CODE:
2316 011022 012710 010112          MOV      #010112,(R0)   ;010112='MOV R1,(R2)''
2317 011026 012760 005012 000002  MOV      #005012,2(R0) ;005012='CLR (R2)''
2318 011034 012760 000207 000004  MOV      #000207,4(R0) ;000207='RTS PC''
2319
2320 011042 012701 000002          MOV      #2,R1         ;SET UP THE REGISTERS USED
2321 011046 012702 177750          MOV      #MAINT,R2     ;IN THE TEST INSTRUCTIONS.
2322
2323 011052 012737 011072 000114  MOV      #RR6,@#CACHVEC ;SET UP THE PARITY TRAP
2324 011060 000240                NOP                    ;VECTOR.
2325 011062 004710                JSR      PC,(R0) ;AND GO.
2326
2327
2328 011064          RR5:                    ;NO TRAP OR ABORT OCCURRED.
2329                                ;MAINTENANCE FUNCTION FOR
2330 011064 104030          1$:      ERROR 30                ;FORCING BAD PARITY ON
2331 011066 000137 011206          JMP      RR9           ;THE MAIN MEMORY ADDRESS
2332                                ;AND CONTROL LINES FAILED.
2333                                ;COME HERE WHEN THE FORCED ERROR OCCURS.
2334 011072          RR6:
2335                                ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2336                                ;COMPARE THE HIGH ORDER
2337 011072 023737 011306 177742  CMP      RRADR2+2,LOADRS+2 ;PARTS OF RRADR2 AND ARG2.
2338 011100 001006          BNE      64$          ;COMPARE THE LOW ORDER
2339 011102 023737 011304 177740  CMP      RRADR2,LOADRS   ;PARTS.
2340
2341 011110 001002          BNE      64$
2342
2343
2344
2345 011112 000137 011130          JMP      RR7           ;THEY WERE EQUAL.
2346
2347 011116 103402          64$:      BLO      65$
2348 011120 000137 011150          JMP      RR8           ;THE FIRST ADDRESS IS LARGER
2349                                ;THAN THE SECOND.
2350 011124 000137 011150          65$:      JMP      RR8           ;THE FIRST IS LESS THAN THE
2351                                ;SECOND.
2352
2353
2354 011130 022626          RR7:      CMP      (SP)+,(SP)+
2355 011132 005726          TST      (SP)+        ;RESTORE THE STACK.
2356 011134 022626          CMP      (SP)+,(SP)+
2357 011136 012737 177777 177744  MOV      #-1,@#MEMERR  ;CLEAR THE CACHE ERROR REGISTER.
2358 011144 000137 011206          JMP      RR9
2359
2360 011150 013737 177744 001630  RR8:      MOV      @#MEMERR,$TMP1
2361 011156 013737 177740 001634  MOV      @#LOADRS,$TMP3
2362 011164 013737 177742 001636  MOV      @#HIADRS,$TMP4
    
```

```
2363 011172 005726          TST      (SP)+
2364 011174 022626          CMP      (SP)+,(SP)+
2365 011176 104031          ERROR   31
2366 011200 012737 000001 177744  RR9:    MOV      #1,@MEMERR          ;CLEAR THE ERROR REGISTER.
2367 011206 005037 177572          CLR      @MMR0              ;TURN OFF MEMORY MANAGEMENT.
2368 011212 005037 172516          CLR      @MMR3
2369 011216 005737 011310          TST      RRADR3            ;GET READY TO GENERATE THE
2370 011222 001007          BNE      RR10              ;NEXT ADDRESS TO BE TESTED.
2371 011224 005737 011310          TST      RRADR3
2372 011230 001004          BNE      RR10
2373 011232 012737 000002 011310  MOV      #2,RRADR3
2374 011240 000415          BR       RR12
2375
2376 011242 006337 011310  RR10:   ASL      RRADR3
2377 011246 006137 011312          ROL      RRADR3+2
2378 011252 000410          BR       RR12
2379
2380 011254 006337 011300  RR11:   ASL      RRADR1
2381 011260 006137 011302          ROL      RRADR1+2
2382 011264 005037 011310          CLR      RRADR3
2383 011270 005037 011312          CLR      RRADR3+2
2384
2385 011274 000137 010610  RR12:   JMP      RR1
2386
2387 011300 000000  RRADR1: .WORD   0          ;3 DOUBLE WORD LOCATIONS
2388 011302 000000          .WORD   0          ;USED TO STORE 22-BIT
2389 011304 000000  RRADR2: .WORD   0          ;ADDRESSES.
2390 011306 000000          .WORD   0
2391 011310 000000  RRADR3: .WORD   0
2392 011312 000000          .WORD   0
2393
2394 011314 104416  RRDONE: RSET              ;DONE!
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409 011316 000004  TST10: SCOPE
2410 011320 012737 000020 001676  MOV      #20,$TIMES          ;;DO 20 ITERATIONS
2411
2412          SS=$TN-1
2413 011326 012737 012204 054230  MOV      #TST11,SKAD        ;SET THE SKAD REGISTER
2414
2415
2416
2417
2418
2415 011334 113737 001502 001626  MOV      $STNM,$TMP0        ;IN CASE THE TEST ABORTS.
2416 011342 012737 054076 000114  MOV      #SPUR,@#CACHVEC    ;INITIALLY EXPECT NO ERRORS
2417 011350 104422
2418
```

```

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) K 6 16-MAY-79 09:11 PAGE 50
CEKBDD.P11 16-MAY-79 08:58 T10 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ZEROES SEQ 0075

2419 011352 012700 172340 MOV #KIPAR0,R0 ;INITIALLY PUT MEMORY
2420 011356 012701 077406 MOV #77406,R1 ;MANAGEMENT IN A 'PASSIVE'
2421 011362 012702 172300 MOV #KIPDR0,R2 ;STATE, THAT IS MAP ALL
2422 011366 012703 000010 MOV #10,R3 ;VIRTUAL ADDRESSES ON TO
2423 011372 010122 64$: MOV R1,(R2)+ ;THEMSELVES AS PHYSICAL
2424 011374 077302 SOB R3,64$ ;ADDRESSES.
2425 011376 005020 CLR (R0)+
2426 011400 012720 000200 MOV #200,(R0)+
2427 011404 012720 000400 MOV #400,(R0)+
2428 011410 012720 000600 MOV #600,(R0)+
2429 011414 012720 001000 MOV #1000,(R0)+
2430 011420 012720 001200 MOV #1200,(R0)+
2431 011424 012720 001400 MOV #1400,(R0)+
2432 011430 012710 177600 MOV #177600,(R0)
2433
2434 011434 104424 SIZE ;GET THE MEMORY SIZE.
2435 011436 000000 SSLOAD: .WORD 0 ;22-BIT ADDRESS OF THE
2436 011440 000000 SSHIAD: .WORD 0 ;HIGHEST WORD IN MEMORY.
2437 011442 042737 000002 011436 BIC #2,SSLOAD ;GET THE HIGHEST WORD MINUS TWO.
2438
2439 011450 012737 000014 177746 MOV #MOM1,@#CONTRL
2440
2441 011456 012737 177776 012162 SS1: MOV #177776,SSADR1 ;INITIALIZE
2442 011464 005037 012164 CLR SSADR1+2
2443 011470 012704 000016 MOV #16,R4
2444 011474 000410 BR SS3
2445
2446 011476 005204 SS2: INC R4 ;TURN ON THE NEXT BIT
2447 011500 052737 000001 012162 BIS #1,SSADR1 ;IN THE FIELD OF ONES
2448 011506 006337 012162 ASL SSADR1
2449 011512 006137 012164 ROL SSADR1+2
2450
2451 011516 012737 000002 012172 SS3: MOV #2,SSMASK ;INITIALIZE THE MASK USER
2452 011524 005037 012174 CLR SSMASK+2 ;TO CREATE THE ZERO IN
2453 ;IN FIELD OF ONES
2454 011530 010405 MOV R4,R5
2455 011532 012737 011540 001512 MOV #SS4,$LPERR
2456
2457 011540 013737 012162 012166 SS4: MOV SSADR1,SSADR2 ;DETERMINE THE TEST ADDRESS.
2458 011546 013737 012164 012170 MOV SSADR1+2,SSADR2+2
2459 011554 043737 012172 012166 BIC SSMASK,SSADR2
2460 011562 043737 012174 012170 BIC SSMASK+2,SSADR2+2
2461
2462 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2463 011570 023737 012170 012200 CMP SSADR2+2,SSCNST+2 ;COMPARE THE HIGH ORDER
2464 011576 001006 BNE 64$ ;PARTS OF SSADR2 AND ARG2.
2465 011600 023737 012166 012176 CMP SSADR2,SSCNST ;COMPARE THE LOW ORDER
2466
2467 011606 001002 BNE 64$ ;PARTS.
2468
2469
2470
2471 011610 000137 011626 JMP SS5 ;THEY WERE EQUAL.
2472
2473 011614 103402 64$: BLO 65$
2474 011616 000137 011626 JMP SS5 ;THE FIRST ADDRESS IS LARGER

```

```

2475
2476 011622 000137 012120      65$:  JMP      SS10      ;THAN THE SECOND!
2477                                ;THE FIRST IS LESS THAN THE
2478                                ;SECOND.
2479 011626                      SS5:
2480                                ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2481                                CMP      SSADR2+2,SSLOAD+2 ;COMPARE THE HIGH ORDER
2482 011626 023737 012170 011440      BNE      64$           ;PARTS OF SSADR2 AND ARG2.
2483 011634 001006                                CMP      SSADR2,SSLOAD ;COMPARE THE LOW ORDER
2484 011636 023737 012166 011436      BNE      64$           ;PARTS.
2485
2486 011644 001002
2487
2488
2489
2490 011646 000137 011664          JMP      SS6           ;THEY WERE EQUAL.
2491
2492 011652 103402      64$:  BLO      65$
2493 011654 000137 012120          JMP      SS10         ;THE FIRST ADDRESS IS LARGER
2494                                ;THAN THE SECOND!
2495 011660 000137 011664      65$:  JMP      SS6           ;THE FIRST IS LESS THAN THE
2496                                ;SECOND.
2497
2498
2499 011664                      SS6:
2500                                ;CONVERT THE PHYSICAL 22-BIT, ADDRESS IN SSADR2 TO A VIRTUAL ADDRESS
2501                                ;WHICH WILL RELOCATE THROUGH KIPAR6 TO THE UNIBUS, THEN THROUGH
2502                                ;THE MAPPING BOX TO THE UNIBUS INPUTS OF THE CACHE AMX.
2503 011664 013737 012166 170200      MOV      SSADR2,@MAPL00 ;SET UP THE MAP REGISTER 0.
2504 011672 013737 012170 170202      MOV      SSADR2+2,@MAPH00
2505 011700 162737 000002 170200      SUB      #2,@MAPL00
2506 011706 005637 170202          SBC      @MAPH00
2507
2508 011712 012700 140000          MOV      #140000,R0    ;A VIRTUAL ADDRESS WHICH WILL
2509                                ;RELOCATE THROUGH KIPAR6.
2510 011716 012737 170000 172354      MOV      #170000,@KIPAR6;RELOCATE TO UNIBUS BASE
2511                                ;ADDRESS OF 000000.
2512 011724 012737 000060 172516      MOV      #60,@MMR3    ;TURN ON THE MAPPING BOX AND
2513                                ;22-BIT MODE.
2514 011732 012737 000001 177572      MOV      #1,@MMR0     ;TURN ON MEMORY MANAGEMENT.
2515                                ;SET UP THE TEST CODE:
2516 011740 012710 010112          MOV      #010112,(R0) ;010112='MOV R1,(R2)'
2517 011744 012760 005012 000002      MOV      #005012,2(R0);005012='CLR (R2)'
2518 011752 012760 000207 000004      MOV      #000207,4(R0);000207='RTS PC'
2519
2520 011760 012701 000002          MOV      #2,R1        ;SET UP THE REGISTERS USED
2521 011764 012702 177750          MOV      #MAINT,R2    ;IN THE TEST INSTRUCTIONS.
2522
2523 011770 012737 012044 000114      MOV      #SS8,@CACHVEC;SET UP THE PARITY TRAP
2524 011776 000240                                NOP                    ;VECTOR.
2525 012000 004710          JSR      PC,(R0) ;AND GO.
2526
2527                                ;NO TRAP OCCURRED.
2528 012002 104032      1$:  ERROR  32
2529 012004 000445          BR      SS10
2530                                ;TRAP TO HERE WHEN THE ERROR OCCURS.
    
```

```

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) M 6 16-MAY-79 09:11 PAGE 52
CEKBD.D.P11 16-MAY-79 08:58 T10 CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FLOATING ZEROES SEQ 0077

2531 012006 SS7:
2532
2533 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2534 012006 023737 012170 177742 CMP SSADR2+2,LOADRS+2 ;COMPARE THE HIGH ORDER
2535 012014 001006 BNE 64$ ;PARTS OF SSADR2 AND ARG2.
2536 012016 023737 012166 177740 CMP SSADR2,LOADRS ;COMPARE THE LOW ORDER
2537
2538 012024 001002 BNE 64$ ;PARTS.
2539
2540
2541
2542 012026 000137 012044 JMP SS8 ;THEY WERE EQUAL.
2543
2544 012032 103402 64$: BLO 65$
2545 012034 000137 012062 JMP SS9 ;THE FIRST ADDRESS IS LARGER
2546 ;THAN THE SECOND!
2547 012040 000137 012062 65$: JMP SS9 ;THE FIRST IS LESS THAN THE
2548 ;SECOND.
2549
2550
2551 012044 022626 SS8: CMP (SP)+,(SP)+
2552 012046 005726 TST (SP)+ ;RESTORE THE STACK
2553 012050 022626 CMP (SP)+,(SP)+
2554 012052 012737 177777 177744 MOV #-1,@MEMERR ;CLEAR THE CACHE ERROR
2555 012060 000417 BR SS10 ;REGISTER.
2556
2557 012062 013737 177744 001630 SS9: MOV @MEMERR,$TMP1 ;REPORT A VALID TEST FAILURE.
2558 012070 013737 177740 001634 MOV @LOADRS,$TMP3
2559 012076 013737 177742 001636 MOV @HIADRS,$TMP4
2560 012104 005726 TST (SP)+
2561 012106 022626 CMP (SP)+,(SP)+
2562 012110 104033 ERROR 33
2563 012112 012737 177777 177744 MOV #-1,@MEMERR
2564
2565 012120 006337 612172 SS10: ASL SSMASK ;ROTATE MASK TO FLOAT 0
2566 012124 006137 012174 ROL SSMASK+2 ;TO THE LEFT.
2567 012130 005305 DEC R5
2568 012132 001402 BEQ 1$
2569 012134 000137 011540 JMP SS4
2570 012140 005037 177572 1$: CLR @MMR0 ;TURN OF MEMORY MANAGEMENT
2571 012144 005037 172516 CLR @MMR3 ;AND THE MAPPING BOX.
2572 012150 020427 000025 CMP R4,#25 ;IS THE TEST DONE?
2573 012154 002012 BGE SS11 ;YES
2574 012156 000137 011476 JMP SS2 ;NO
2575
2576 012162 000000 SSADR1: .WORD 0 ;USED TO GENERATE THE
2577 012164 000000 .WORD 0 ;TEST ADDRESSES.
2578 012166 000000 SSADR2: .WORD 0
2579 012170 000000 .WORD 0
2580 012172 000000 SSMASK: .WORD 0
2581 012174 000000 .WORD 0
2582
2583 012176 124704 SSCNST: .WORD BOTPRG ;CONTAINS THE ADDRESS OF
2584 012200 000000 .WORD 0 ;THE LAST WORD OF THIS PROGRAM.
2585
2586 012202 104416 SS11: RSET ;DONE.

```

2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642

012204 000004
012206 012737 000004 001676
000011
012214 012737 012442 054230
012222 113737 001502 001626
012230 012737 054076 000114
012236 012737 000014 177746
012244 104424
012246 000000
012250 000000
012252 012737 157776 012436
012260 005737 012250
012264 001007
012266 023737 012436 012246
012274 003403
012276 013737 012246 012436
012304 012700 124704
012310 010020
012312 020037 012436
012316 101774
012320 012700 124704
012324 011001
012326 020001
012330 001411
012332 010037 001640
012336 010137 001632
012342 010037 001634
012346 005037 001636
012352 104034
012354 005120
012356 020037 012436
012362 101760
012364 012700 124704
012370 011001
012372 010002
012374 005102

```

*****
*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.
*
*****
TST11:  SCOPE
        MOV      #4,$TIMES      ;;DO 4 ITERATIONS
PP=$TN-1
        ;;SET THE SKAD REGISTER
        MOV      #TST12,SKAD    ;IN CASE THE TEST ABORTS.
        MOVB     $STNM,$STMP0
        MOV      #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
PP1:    MOV      #M1MO,@#CONTRL ;FORCE MISSES TO BOTH GROUPS
        SIZE
PLOAD:  .WORD   0              ;LOW ORDER 16-BITS AND
PFIAD:  .WORD   0              ;HIGH ORDER 6-BITS OF THE
        ;HIGHEST WORD ADDRESS IN
        ;MEMORY.
        MOV      #157776,PPLIM  ;ESTABLISH THE UPPER LIMIT
        TST     PFIAD          ;FOR THE TEST.
        BNE     PP2
        CMP     PPLIM,PLOAD
        BLE     PP2
        MOV     PLOAD,PPLIM
PP2:    MOV      #BOTPRG,R0     ;THE LOW LIMIT FOR THIS TEST.
1$:    MOV      R0,(R0)+       ;WRITE THE ADDRESS IN THE
        CMP     R0,PPLIM      ;ADDRESS.
        BLOS   1$
PP3:    MOV      #BOTPRG,R0
        MOV     (R0),R1
        CMP     R0,R1
        BEQ    PP4
        MOV     R0,$TMP5
        ;REPORT ERROR.
        MOV     R1,$TMP2
        MOV     R0,$TMP3
        CLR    $TMP4
1$:    ERROR   34
PP4:    COM     (R0)+
        CMP     R0,PPLIM
        BLOS   PP3
PP5:    MOV      #BOTPRG,R0     ;GO BACK AND CHECK
        MOV     (R0),R1        ;THE COMPLIMENTED PATTERNS.
        MOV     R0,R2
        COM     R2
    
```

```

2643 012376 020102      CMP      R1,R2
2644 012400 001411      BEQ      PP6
2645 012402 010237 001640      MOV      R2,$TMP5
2646 012406 010137 001632      MOV      R1,$TMP2
2647 012412 010037 001634      MOV      R0,$TMP3
2648 012416 005037 001636      CLR      $TMP4
2649 012422 104034      'S:      ERROR 34
2650
2651 012424 005120      PP6:     COM      (R0)+
2652 012426 020037 012436      CMP      R0,PPLIM
2653 012432 001356      BNE      PP5
2654 012434 000401      BR       PP7
2655
2656 012436 000000      PPLIM:  .WORD 0
2657
2658 012440 104416      PP7:     RSET          ;DONE!
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669 012442 000004      ;*****
2670 012444 012737 000002 001676      ;*TEST 12      CACHE ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS DUAL ADDRESS TEST
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698

```

```

;*****
;*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
;*BOX HERE THEREBY TESTING THE UNIBUS INPUTS TO THE AMX.
;*
;*****
TST12:  SCOPE
        MOV      #2,$TIMES          ;;DO 2 ITERATIONS
TT=$TN-1
        MOV      #TST13,SKAD       ;SET THE SKAD REGISTER
                                        ;IN CASE THE TEST ABORTS.
        MOVB     $TSTNM,$TMP0
        MOV      #SPUR,@#CACHVEC   ;EXPECT NO PARITY ERRORS.
TT1:    MOV      #M1M0,@#CONTRL    ;FORCE MISSES TO BOTH GROUPS.
        SIZE
TTLOAD: .WORD 0                    ;DETERMINE THE HIGHEST
TTHIAD: .WORD 0                    ;WORD IN MEMORY.
        MOV      #157776,TTLIM     ;DETERMINE THE UPPER LIMIT
        TST      TTHIAD            ;FOR THE TEST.
        BNE     TT2
        CMP      TTLIM,TTLOAD
        BLE     TT2
TT2:    MOV      TTLOAD,TTLIM
        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)+

```



```

2699 012576 012720 000400      MOV      #400,(R0)+
2700 012602 012720 000600      MOV      #600,(R0)+
2701 012606 012720 001000      MOV      #1000,(R0)+
2702 012612 012720 001200      MOV      #1200,(R0)+
2703 012616 012720 001400      MOV      #1400,(R0)+
2704 012622 012710 177600      MOV      #177600,(R0)
2705
2706 012626 012737 000060 172516      MOV      #60,@MMR3      ;TURN ON MEMORY MANAGEMENT.
2707 012634 012737 000001 177572      MOV      #1,@MMR0
2708 012642 012700 124704      MOV      #BOTPRG,R0      ;INITIALIZE A POINTER.
2709
2710 012646                      1$:
2711
2712 012646 010037 170200      MOV      R0,@MAPL00      ;RELOCATE THE ADDRESS IN
2713 012652 005037 170202      CLR      @MAPH00          ;R0 TO THE UNIBUS,
2714 012656 012737 170000 172354      MOV      #170000,@KIPAR6;THROUGH THE MAPPING BOX
2715 012664 012701 140000      MOV      #140000,R1      ;TO THE CACHE.
2716
2717
2718 012670 010011      MOV      R0,(R1)          ;WRITE THE ADDRESS IN THE
2719 012672 062700 000002      ADD      #2,R0            ;ADDRESS
2720 012676 020037 013076      CMP      R0,TTLIM
2721 012702 101761      BLOS
2722
2723 012704 012700 124704      MOV      #BOTPRG,R0
2724
2725 012710                      TT3:
2726
2727 012710 010037 170200      MOV      R0,@MAPL00      ;RELOCATE THE ADDRESS IN
2728 012714 005037 170202      CLR      @MAPH00          ;R0 TO THE UNIBUS,
2729 012720 012737 170000 172354      MOV      #170000,@KIPAR6;THROUGH THE MAPPING BOX
2730 012726 012701 140000      MOV      #140000,R1      ;TO THE CACHE.
2731
2732
2733 012732 011102      MOV      (R1),R2          ;READ BACK THE ADDRESS
2734 012734 020002      CMP      R0,R2            ;AS DATA IN THE LOCATION
2735 012736 001411      BEQ      TT4              ;IT ADDRESSES.
2736 012740 010037 001640      MOV      R0,$TMP5        ;REPORT ERROR IF NOT
2737                                ;EQUAL.
2738 012744 010237 001632      MOV      R2,$TMP2
2739 012750 010037 001634      MOV      R0,$TMP3
2740 012754 005037 001636      CLR      $TMP4
2741 012760 104035      ERROR   35
2742 012762 005111      COM      (R1)              ;WRITE BACK THE
2743 012764 062700 000002      ADD      #2,R0            ;COMPLIMENTED DATA.
2744 012770 020037 013076      CMP      R0,TTLIM
2745 012774 101745      BLOS    TT3
2746
2747 012776 012700 124704      MOV      #BOTPRG,R0
2748
2749 013002                      TT5:
2750
2751 013002 010037 170200      MOV      R0,@MAPL00      ;RELOCATE THE ADDRESS IN
2752 013006 005037 170202      CLR      @MAPH00          ;R0 TO THE UNIBUS,
2753 013012 012737 170000 172354      MOV      #170000,@KIPAR6;THROUGH THE MAPPING BOX
2754 013020 012701 140000      MOV      #140000,R1      ;TO THE CACHE.
    
```

```

2755
2756
2757 013024 011102      MOV      (R1),R2      ;GO BACK AND CHECK
2758 013026 010003      MOV      R0,R3       ;THE COMPLIMENTED PATTERNS.
2759 013030 005103      COM      R3
2760 013032 020203      CMP      R2,R3
2761 013034 001411      BEQ      TT6
2762 013036 010337 001640  MOV      R3,$TMP5     ;REPORT ERROR
2763 013042 010237 001637  MOV      R2,$TMP2
2764 013046 010037 001634  MOV      R0,$TMP3
2765 013052 005037 001636  CLR      $TMP4
2766 013056 104035      1$:      ERROR      35
2767
2768 013060 005111      TT6:     COM      (R1)      ;COMPLIMENT BACK THE DATA.
2769 013062 062700 000002  ADD      #2,R0
2770 013066 020037 013076  CMP      R0,TTLIM
2771 013072 001343      BNE      TT5
2772 013074 000401      BR       TT7
2773
2774 013076 000000      TTLIM:   .WORD    0
2775
2776 013100 104416      TT7:     RSET          ;DONE.
2777
2778
2779

```

```

:*****
:*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.
:*
:*****

```

```

2800
2801 013102 000004      TST13:  SCOPE
2802 013104 012737 000040 001676  MOV      #40,$TIMES    ;;DO 40 ITERATIONS
2803
2804 013112 012737 014254 054230  MOV      #TST14,SKAD   ;SET THE SKAD REGISTER
2805
2806 013120 113737 001502 001626  MOV      $TSTM,$TMP0   ;IN CASE THE TEST ABORTS.
2807 013126 012737 054076 000114  MOV      #SPUR,@#CACHVEC
2808
2809 013134 104422      MMSKIP          ;SEE IF THE SWITCH REGISTER
2810

```

```

;REFLECTS THE USERS DESIRE

```

```

2811                                     ;TO ELIMINATE EXECUTION OF ANY TESTS
2812                                     ;USING MEMORY MANAGEMENT. IF
2813                                     ;SO GO TO THE NEXT TEST.
2814
2815 013136 012700 172340                MOV    #KIPAR0,R0                ;INITIALLY PUT MEMORY
2816 013142 012701 077406                MOV    #77406,R1                ;MANAGEMENT IN A 'PASSIVE'
2817 013146 012702 172300                MOV    #KIPDR0,R2               ;STATE, THAT IS MAP ALL
2818 013152 012703 000010                MOV    #10,R3                   ;VIRTUAL ADDRESSES ON TO
2819 013156 010122                        64$: MOV    R1,(R2)+                ;THEMSELVES AS PHYSICAL
2820 013160 077302                        SOB    R3,64$                   ;ADDRESSES.
2821 013162 005020                        CLR    (R0)+
2822 013164 012720 000200                MOV    #200,(R0)+
2823 013170 012720 000400                MOV    #400,(R0)+
2824 013174 012720 000600                MOV    #600,(R0)+
2825 013200 012720 001000                MOV    #1000,(R0)+
2826 013204 012720 001200                MOV    #1200,(R0)+
2827 013210 012720 001400                MOV    #1400,(R0)+
2828 013214 012710 177600                MOV    #177600,(R0)
2829
2830
2831 013220 104424                        SIZE
2832 013222 000000                        ZADLO: .WORD 0                  ;THE HIGHEST ADDRESSABLE
2833 013224 000000                        ZADHI: .WORD 0                  ;MEMORY WORD AVAILABLE.
2834
2835 013226 005037 014020                CLR    ZFLG1                    ;ZFLG1 INDICATES WHICH GROUP
2836                                     ;IS BEING TESTED.
2837                                     ;ZFLG1 = 0, TESTING GROUP 0.
2838                                     ;ZFLG1 = 1, TESTING GROUP 1.
2839                                     ;TEST GROUP 0 FIRST.
2840
2841 013232 012737 000030 014026          MOV    #SOM1,ZGS                 ;ZGS AND ZGM CONTAIN
2842 013240 012737 000044 014024          MOV    #S1M0,ZGM                 ;PATTERNS TO BE USED IN
2843                                     ;THE CACHE CONTROL REGISTER.
2844 013246 005037 014022                CLR    ZFLG2                    ;ZFLG2 INDICATES WHICH
2845                                     ;4-BIT ADDRESS FIELD, OR
2846                                     ;WHICH COMPARATOR, IS
2847                                     ;BEING TESTED.
2848                                     ;ZFLG2 = 0, BITS 10 THROUGH 13
2849                                     ;ZFLG2 = 1, BITS 14 THROUGH 17
2850                                     ;ZFLG2 = 2, BITS 18 THROUGH 21
2851                                     ;ZFLG2 = 3, DONE.
2852
2853 013252 005737 014022                Z*:  TST    ZFLG2                ;SEE WHICH COMPARATOR
2854 013256 001010                        BNE    Z2                        ;IS BEING TESTED ON THIS
2855                                     ;PASS AND PUT THE SIXTEEN
2856                                     ;POSSIBLE ADDRESSES NEEDED
2857                                     ;FOR THE TEST IN ZTABLE.
2858
2858 013260 012737 002000 014046          MOV    #2000,ZTABLE+4           ;BITS 10-13
2859 013266 005037 014050                CLR    ZTABLE+6
2860 013272 004737 014144                JSR    PC,ZCMTBL                ;CALL ZCMTBL TO FINISH THE TABLE.
2861 013276 000432                        BR     Z5
2862
2863 013300 022737 000001 014022          Z2:  CMP    #1,ZFLG2
2864 013306 001010                        BNE    Z3
2865
2866 013310 012737 040000 014046          MOV    #40000,ZTABLE+4         ;BITS 14-17
    
```

```

2867 013316 005037 014050 CLR ZTABLE+6
2868 013322 004737 014144 JSR PC,ZCMTBL ;GET ZCMTBL TO FINISH SETTING
2869 013326 000416 BR Z5 ;UP THE TABLE.
2870
2871 013330 022737 000002 014022 Z3: CMP #2,ZFLG2
2872 013336 001010 BNE Z4
2873
2874 013340 012737 000004 014050 MOV #4,ZTABLE+6 ;BITS 18-21
2875 013346 005037 014046 CLR ZTABLE+4
2876 013352 004737 014144 JSR PC,ZCMTBL
2877 013356 000402 BR Z5
2878
2879 013360 000137 013752 Z4: JMP Z14 ;DONE WITH THIS GROUP.
2880
2881 013364 012701 014032 Z5: MOV #ZTHR,R1
2882 013370 013737 014024 177746 MOV ZGM,@#CONTRL
2883 013376 005711 TST (R1) ;MAKE ZTHR A HIT IN BOTH GROUPS.
2884 013400 013737 014026 177746 MOV ZGS,@#CONTRL
2885 013406 005711 TST (R1)
2886
2887 ;FROM NOW ON SELECT THE GROUP BEING TESTED
2888 ;WHILE MISSING THE OTHER GROUP.
2889 013410 012737 000020 172516 MOV #20,@MMR3 ;TURN ON MEMORY MANAGEMENT.
2890 013416 012737 000001 177572 MOV #1,@MMR0 ;22-BIT MODE!
2891
2892 013424 012701 014042 MOV #ZTABLE,R1 ;INITIALIZE R1 AS A POINTER
2893 ;TO THE ADDRESS WHICH WILL
2894 ;BE MADE A HIT.
2895
2896 013430 Z7:
2897 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2898
2899
2900
2901 013430 023761 013224 000002 CMP ZADLO+2,2(R1) ;COMPARE THE HIGH ORDER
2902 013436 001005 BNE 64$ ;PARTS OF ZADLO AND (R1).
2903 013440 023711 013222 CMP ZADLO,(R1) ;THEN IF NECESSARY
2904 013444 001002 BNE 64$ ;COMPARE THE LOW ORDER PARTS.
2905
2906 013446 000137 013464 JMP 1$ ;THEY WERE EQUAL!
2907
2908 013452 103402 64$: BLO 65$
2909 013454 000137 013464 JMP 1$ ;THE FIRST ADDRESS IS LARGER
2910 ;THAN THE SECOND.
2911 013460 000137 013752 65$: JMP Z14 ;THE FIRST IS LESS THAN THE
2912 ;SECOND.
2913
2914
2915 013464 012702 014042 1$: MOV #ZTABLE,R2 ;INITIALIZE A POINTER TO
2916 ;THE ADDRESSES WHICH WILL
2917 ;BE FED THROUGH THE COMPARATOR
2918 ;AGAINST THE ADDRESS POINTED
2919 ;TO BY THE OTHER POINTER, R1
2920
2921 013470 020102 Z8: CMP R1,R2 ;DON'T TEST THE ADDRESS
2922 013472 001511 BEQ Z12 ;AGAINST ITSELF HERE.
    
```

```

2923
2924 013474          79:
2925
2926          ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
2927
2928
2929 013474 023762 013224 000002      CMP      ZADLO+2,2(R2)      ;COMPARE THE HIGH ORDER
2930 013502 001005          BNE      64$              ;PARTS OF ZADLO AND (R2).
2931 013504 023712 013222          CMP      ZADLO,(R2)       ;THEN IF NECESSARY
2932 013510 001002          BNE      64$              ;COMPARE THE LOW ORDER PARTS.
2933
2934 013512 000137 013530          JMP      Z10              ;THEY WERE EQUAL.
2935
2936 013516 103402          64$:    BLO      65$              ;THE FIRST ADDRESS IS LARGER
2937 013520 000137 013530          JMP      Z10              ;THAN THE SECOND!
2938
2939 013524 000137 013730          65$:    JMP      Z13              ;THE FIRST IS LESS THAN THE
2940
2941
2942
2943 013530          Z10:
2944
2945 013530 011103          MOV      (R1),R3          ;GET THE PHYSICAL ADDRESS POINTED
2946 013532 042703 177700          BIC      #177700,R3      ;TO BY R1 AND ESTABLISH
2947 013536 011105          MOV      (R1),R5 ;A VIRTUAL ADDRESS WHICH
2948 013540 016104 000002          MOV      2(R1),R4        ;WILL RELOCATE THROUGH
2949 013544 073427 177772          ASHC    #-6,R4          ;KIPAR6. SETUP KIPAR6 AND
2950 013550 010537 172354          MOV      R5,@#KIPAR6    ;LEAVE THE VIRTUAL ADDRESS
2951 013554 062703 140000          ADD      #140000,R3     ;IN R3.
2952
2953
2954 013560 005713          TST      (R3)
2955 013562 005713          TST      (R3)           ;SEE IF YOU CAN GET A HIT.
2956 013564 032737 000010 177752          BIT      #10,@#HITMIS
2957 013572 001011          BNE      Z11
2958 013574 013737 014020 001630          MOV      ZFLG1,$TMP1    ;NO! REPORT THE FAILURE
2959 013602 011137 001632          MOV      (R1),$TMP2
2960 013606 016137 000002 001634          MOV      2(R1),$TMP3
2961 013614 104026          1$:    ERROR 26
2962
2963 013616          Z11:
2964
2965 013616 011203          MOV      (R2),R3          ;GET THE PHYSICAL ADDRESS POINTED
2966 013620 042703 177700          BIC      #177700,R3      ;TO BY R2 AND ESTABLISH
2967 013624 011205          MOV      (R2),R5 ;A VIRTUAL ADDRESS WHICH
2968 013626 016204 000002          MOV      2(R2),R4        ;WILL RELOCATE THROUGH
2969 013632 073427 177772          ASHC    #-6,R4          ;KIPAR6. SETUP KIPAR6 AND
2970 013636 010537 172354          MOV      R5,@#KIPAR6    ;LEAVE THE VIRTUAL ADDRESS
2971 013642 062703 140000          ADD      #140000,R3     ;IN R3.
2972
2973
2974 013646 000240          NOP
2975 013650 005713          TST      (R3)           ;FOR SCOPING WITH AN OSCILLOSCOPE.
2976 013652 032737 000010 177752          BIT      #10,@#HITMIS   ;MAKE SURE THERE IS NO
2977 013660 001416          BEQ      Z12              ;MATCH. A MISS?
2978 013662 013737 014020 001630          MOV      ZFLG1,$TMP1    ;GOT A HIT! SO REPORT
    
```

H 7

CEKBU-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY1: 30A(1052) 16-MAY-79 09:11 PAGE 60
 CEKBDD.P11 16-MAY-79 08:58 T13 CACHE ADDRESS MEMORY COMPARATOR TEST SEQ 0085

```

2979 013670 011137 001632      MOV      (R1), $TMP2      ;FAILURE
2980 013674 016137 000002 001634  MOV      2(R1), $TMP3
2981 013702 011237 001636      MOV      (R2), $TMP4
2982 013706 016237 000002 001640  MOV      2(R2), $TMP5
2983 013714 104027      18:      ERROR      27
2984
2985 013716 062702 000004      Z12:     ADD      #4, R2      ;MOVE POINTER TO NEXT AMX
2986      ;SIDE COMPARATOR INPUT ADDRESS.
2987 013722 020227 014142      CMP      R2, #ZTABOT     ;DONE?
2988 013726 001260      BNE      Z8              ;BRANCH IF NOT DONE.
2989
2990 013730 062701 000004      Z13:     ADD      #4, R1      ;GO TO THE NEXT ADDRESS
2991 013734 020127 014142      CMP      R1, #ZTABOT     ;IN THE TABLE; OR IS THE
2992 013740 001233      BNE      Z7              ;TEST USING THIS ADDRESS TABLE DONE?
2993      ;IF NOT GO TO Z7.
2994 013742 005237 014022      INC      ZFLG2           ;IF DONE WITH THESE ADDRESSES
2995 013746 000137 013252      JMP      Z1              ;GO BACK TO COMPUTE THE
2996      ;NEXT ADDRESS TABLE, THAT IS
2997      ;CHECK THE NEXT 4-BIT
2998      ;COMPARATOR
2999      ;TURN OFF MEMORY MANAGEMENT.
3000 013752 005037 177572      Z14:     CLR      @AMMR0
3001 013756 005037 172516      CLR      @AMMR3
3002 013762 005737 014020      TST      ZFLG1           ;SEE IF BOTH GROUPS HAVE
3003 013766 001131      BNE      Z15           ;BEEN TESTED. BRANCH IF YES
3004 013770 005237 014020      INC      ZFLG1           ;OTHERWISE CHANGE THE
3005 013774 012737 000044 014026  MOV      #S1MO, ZGS      ;PATTERNS USED IN THE CACHE
3006 014002 012737 000030 014024  MOV      #SOM1, ZGM      ;CONTROL REGISTER AND GO
3007 014010 005037 014022      CLR      ZFLG2           ;BACK TO TEST GROUP 1.
3008 014014 000137 013252      JMP      Z1
3009 014020 000000      ZFLG1:   .WORD      0      ;FLAG WHICH DESIGNATES WHICH
3010      ;GROUP IS BEING TESTED, 0 OR 1.
3011 014022 000000      ZFLG2:   .WORD      0      ;FLAG WHICH DESIGNATES WHICH
3012      ;COMPARATOR IS BEING TESTED:
3013      ;0 - BITS 10 THROUGH 13
3014      ;1 - BITS 14 THROUGH 17
3015      ;2 - BITS 18 THROUGH 21.
3016
3017 014024 000000      ZGM:     .WORD      0      ;PATTERNS USED IN THE HIT
3018 014026 000000      ZGS:     .WORD      0      ;AND MISS REGISTER.
3019 014030 000000      .WORD      0
3020 014032 000000      ZTHR:    .WORD      0
3021 014034 000000      .WORD      0
3022
3023 014036 000000      ZTMP1:   .WORD      0      ;TEMPORARY STORAGE LOCATIONS
3024 014040 000000      ZTMP2:   .WORD      0      ;USED BY THE ROUTINE, ZCMTBL,
3025      ;TO GENERATE THE TEST ADDRESS
3026      ;TABLE, ZTABLE.
3027
3028 014042 000040      ZTABLE:  .BLKW     40      ;THE TEST ADDRESS TABLE.
3029 014142 000000      ZTABOT:  .WORD      0      ;PRECISION, 22-BIT, ADDRESSES.
3030
3031      ;THIS ROUTINE IS CALLED TO GENERATE THE TEST ADDRESS
3032      ;TABLE, BY A 'JSR PC, ZCMTBL'. IT CLEARS THE FIRST
3033      ;ENTRY; IT ASSUMES THE THE BASE ADDRESS HAS BEEN
3034      ;PLACED IN THE SECOND ENTRY BEFORE CONTROL IS PASSED

```

```
3035 ;TO IT; THEN, STARTING WITH THE THIRD ENTRY, IT COMPUTES
3036 ;EACH ENTRY BY ADDING THE BASE ADDRESS TO THE PRECEEDING
3037 ;ENTRY.
3038 014144 012701 014042 ZCMTBL: MOV #ZTABLE,R1 ;ESTABLISH A POINTER TO
3039 ;THE TABLE.
3040 014150 005021 CLR (R1)+ ;CLR THE FIRST ENTRY.
3041 014152 005021 CLR (R1)+
3042 014154 012700 000016 MOV #16,R0
3043 014160 012137 014036 1$: MOV (R1)+,ZTMP1 ;SAVE THE CURRENT ENTRY
3044 014164 012137 014040 MOV (R1)+,ZTMP2
3045 ;ADD THE OFFSET TO THE
3046 ;DOUBLE PRECISION ADDITION, UNSIGNED
3047
3048
3049
3050 014170 013711 014036 MOV ZTMP1,(R1)
3051 014174 013761 014040 000002 MOV ZTMP1+2,2(R1)
3052 014202 063711 014046 ADD ZTABLE+4,(R1)
3053 014206 005561 000002 ADC 2(R1)
3054 014212 063761 014050 000002 ADD ZTABLE+4+2,2(R1)
3055 014220 077021 SOB R0,1$ ;LOOP UNTIL ZTABLE IS FILLED.
3056
3057
3058 014222 012702 000020 MOV #20,R2
3059 014226 012701 014042 MOV #ZTABLE,R1
3060 014232 012700 014032 MOV #ZTHR,R0
3061 014236 042700 176000 BIC #176000,R0
3062 014242 060021 2$: ADD R0,(R1)+
3063 014244 005721 TST (R1)+
3064 014246 077203 SOB R2,2$
3065
3066 014250 000207 RTS PC ;THE RETURN
3067
3068 014252 104416 Z15: RSET ;DONE!
3069
3070
3071 ;*****
3072 ;*TEST 14 CACHE ADDRESS MEMORY COUNT PATTERN TEST
3073 ;*
3074 ;*THIS IS A TEST OF THE ADDRESS MEMORY IN THE CACHE.
3075 ;*EVERY BIT IN THE MEMORY IS TURNED ON AND OFF WITHIN
3076 ;*THE LIMITATIONS OF MEMORY SIZE. THE MANNER IN WHICH
3077 ;*THIS IS DONE IS TO ATTEMPT TO MAKE EVERY ADDRESS
3078 ;*IN AVAILABLE MEMORY A HIT IN EACH GROUP.
3079 ;*
3080 ;*****
3081 014254 000004 TST14: SCOPE
3082 014256 012737 000002 001676 MOV #2,$TIMES ;:DO 2 ITERATIONS
3083 000014 BB-$TN-1
3084 014264 BB0:
3085 ;SET THE SKAD REGISTER
3086 014264 012737 015304 054230 MOV #TST15,SKAD ;IN CASE THE TEST ABORTS.
3087
3088 014272 113737 001502 001626 MOVB $TSTNM,$TMP0
3089
3090 014300 104422 MMSKIP
```

```

3091
3092 014302 104424
3093 014304 000000
3094 014306 000000
3095
3096 014310 005037 015006 CLR BBFLG1 ;TEST GROUP 0 FIRST.
3097 014314 012737 000034 015016 MOV #SOMOM1, BBGS
3098 014322 012737 000054 015020 MOV #S1MOM1, BBGM
3099
3100 014330 012737 054076 000114 BB1: MOV #SPUR, @#CACHVEC ;EXPECT NO ERRORS, FOR NOW.
3101 014336 012700 014264 MOV #BB0, R0 ;MAKE THIS CODE HITS IN
3102 014342 012701 001000 MOV #1000, R1 ;THE GROUP NOT BEING TESTED.
3103 014346 013737 015016 177746 BB2: MOV BBGS, @#CONTRL
3104 014354 005760 002000 TST 2000(R0)
3105 014360 013737 015020 177746 MOV BBGM, @#CONTRL
3106 014366 005720 TST (R0)+
3107 014370 077112 SOB R1, BB2
3108
3109 014372 013700 015016 MOV BBGS, R0 ;FROM NOW ON FORCE
3110 014376 042700 177717 BIC #177717, R0 ;SELECT THE GROUP BEING
3111 014402 010037 177746 MOV R0, @#CONTRL ;TESTED.
3112
3113 014406 012700 014772 BB3: MOV #BBADR1, R0 ;INITIALIZE.
3114 014412 012720 124704 MOV #BOTPRG, (R0)+ ;CONTAINS THE TEST ADDRESS.
3115 014416 005020 CLR (R0)+
3116 014420 005020 CLR (R0)+ ;CONTAINS THE LOGICAL 'OR'
3117 014422 005020 CLR (R0)+ ;OF FAILING ADDRESSES.
3118 014424 012720 177777 MOV #-1, (R0)+ ;CONTAINS THE LOGICAL 'AND'
3119 014430 012720 177777 MOV #-1, (R0)+ ;OF BAD ADDRESSES
3120
3121
3122 014434 012700 172340 MOV #KIPARC, R0 ;INITIALLY PUT MEMORY
3123 014440 012701 077406 MOV #77406, R1 ;MANAGEMENT IN A 'PASSIVE'
3124 014444 012702 172300 MOV #KIPDR0, R2 ;STATE, THAT IS MAP ALL
3125 014450 012703 000010 MOV #10, R3 ;VIRTUAL ADDRESSES ON TO
3126 014454 010122 64$: MOV R1, (R2)+ ;THEMSELVES AS PHYSICAL
3127 014456 077302 SOB R3, 64$ ;ADDRESSES.
3128 014460 005020 CLR (R0)+
3129 014462 012720 000200 MOV #200, (R0)+
3130 014466 012720 000400 MOV #400, (R0)+
3131 014472 012720 000600 MOV #600, (R0)+
3132 014476 012720 001000 MOV #1000, (R0)+
3133 014502 012720 001200 MOV #1200, (R0)+
3134 014506 012720 001400 MOV #1400, (R0)+
3135 014512 012710 177600 MOV #177600, (R0)
3136
3137 014516 012737 000020 172516 MOV #20, @#MMR3 ;TURN ON MEMORY MANAGEMENT.
3138 014524 012737 000001 177572 MOV #1, @#MMR0
3139
3140 014532 005037 015010 CLR BBFLG2 ;INITIALIZE THE ERROR
3141 014536 005037 015012 CLR BBCNT1 ;FLAG AND COUNT.
3142 014542 005037 015014 CLR BBCNT1+2
3143
3144 014546 012737 015022 000114 MOV #BBERR1, @#CACHVEC ;PREPARE FOR ERRORS.
3145
3146 014554 BB4:
    
```



```

3147
3148 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3149 014554 023737 014306 014774 CMP BBLOAD+2, BBADR1+2 ;COMPARE THE HIGH ORDER
3150 014562 001006 BNE 64$ ;PARTS OF BBLOAD AND ARG2.
3151 014564 023737 014304 014772 CMP BBLOAD, BBADR1 ;COMPARE THE LOW ORDER
3152
3153 014572 001002 BNE 64$ ;PARTS.
3154
3155
3156
3157 014574 000137 014612 JMP BB5 ;THEY WERE EQUAL.
3158
3159 014600 103402 64$: BLO 65$
3160 014602 000137 014710 JMP BB7 ;THE FIRST ADDRESS IS LARGER
3161 ;THAN THE SECOND.
3162 014606 000137 014612 65$: JMP BB5 ;THE FIRST IS LESS THAN THE
3163 ;SECOND.
3164
3165
3166 014612 012700 014772 BB5: MOV #BBADR1, R0 ;SET UP MEMORY MANAGEMENT.
3167
3168 014616 011003 MOV (R0), R3 ;GET THE PHYSICAL ADDRESS POINTED
3169 014620 042703 177700 BIC #177700, R3 ;TO BY R0 AND ESTABLISH
3170 014624 011005 MOV (R0), R5 ;A VIRTUAL ADDRESS WHICH
3171 014626 016004 000002 MOV 2(R0), R4 ;WILL RELOCATE THROUGH
3172 014632 073427 177772 ASHC #-6, R4 ;KIPAR6. SETUP KIPAR6 AND
3173 014636 010537 172354 MOV R5, @#KIPAR6 ;LEAVE THE VIRTUAL ADDRESS
3174 014642 062703 140000 ADD #140000, R3 ;IN R3.
3175
3176
3177 014646 000240 NOP ;FOR SCOPING WITH AN OSCILLOSCOPE.
3178 014650 005713 TST (R3) ;TRY TO GET A HIT.
3179 014652 005713 TST (R3)
3180
3181 014654 032737 000010 177752 BIT #10, @#HITMIS ;WAS IT A HIT?
3182 014662 001004 BNE BB6 ;BRANCH IF YES, OTHERWISE
3183 ;REPORT ERROR.
3184 014664 013737 015006 001632 MOV BBFLG1, $TMP2
3185 014672 104036 1$: ERROR 36
3186
3187 014674 062737 000004 014772 BB6: ADD #4, BBADR1 ;MOVE TO NEXT WORD PAIR.
3188 014702 005537 014774 ADC BBADR1+2
3189 014706 000722 BR BB4
3190
3191 014710 005737 015010 BB7: TST BBFLG2 ;DID AN ERROR OCCUR IN
3192 014714 001410 BEQ BB8 ;THAT GROUP, IF YES PRINT
3193 014716 112737 000037 001516 MOVB #37, $ITEMB ;AN ERROR SUMMARY
3194 014724 013737 015006 001630 MOV BBFLG1, $TMP1
3195 014732 004737 054744 JSR PC, ERTYPE
3196
3197 014736 005737 015006 BB8: TST BBFLG1 ;HAVE BOTH GROUPS BEEN TESTED?
3198 014742 001157 BNE BBDONE
3199 014744 012737 000001 015006 MCV #1, BBFLG1 ;IF NOT, GO BACK AND
3200 014752 012737 000054 015016 MOV #S1MOM1, BBGS ;TEST GROUP 1
3201 014760 012737 000034 015020 MOV #S0MOM1, BBGM
3202 014766 000137 014330 JMP BB1
    
```

```

3203
3204 014772 000000 BBADR1: .WORD 0 ;THE TEST ADDRESS.
3205 014774 000000 .WORD 0
3206 014776 000000 BBADR2: .WORD 0 ;LOGICAL 'OR' OF BAD ADDRESSES.
3207 015000 000000 .WORD 0
3208 015002 000000 BBADR3: .WORD 0 ;LOGICAL 'AND' OF BAD ADDRESSES.
3209 015004 000000 .WORD 0
3210
3211 015006 000000 BBFLG1: .WORD 0 ;FLAG: 1, IF TESTING GROUP 1,
3212 .WORD 0 ;OR 0, IF TESTING GROUP 0.
3213 015010 000000 BBFLG2: .WORD 0 ;ERROR FLAG: 0, IF NO ERRORS
3214 .WORD 0 ;OCCURRED IN THE TESTED
3215 .WORD 0 ;GROUP.
3216 015012 000000 BBCNT1: .WORD 0 ;ERROR COUNT.
3217 015014 000000 .WORD 0
3218
3219 015016 000000 BBGS: .WORD 0 ;PATTERNS FOR THE CACHE
3220 015020 000000 BBGM: .WORD 0 ;CONTROL REGISTER
3221
3222 015022 BBERR1:
3223
3224 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3225 015022 023737 177742 014774 CMP LOADRS+2, BBADR1+2 ;COMPARE THE HIGH ORDER
3226 015030 001006 BNE 64$ ;PARTS OF LOADRS AND ARG2.
3227 015032 023737 177740 014772 CMP LOADRS, BBADR1 ;COMPARE THE LOW ORDER
3228
3229 015040 001002 BNE 64$ ;PARTS.
3230
3231
3232
3233 015042 000137 015060 JMP BBERR2 ;THEY WERE EQUAL!
3234
3235 015046 103402 64$: BLO 65$
3236 015050 000137 054076 JMP SPUR ;THE FIRST ADDRESS IS LARGER
3237 ;THAN THE SECOND!
3238 015054 000137 054076 65$: JMP SPUR ;THE FIRST IS LESS THAN THE
3239 ;SECOND.
3240
3241
3242 015060 032737 000060 177744 BBERR2: BIT #60, @MEMERR ;MAKE SURE A CACHE ADDRESS
3243 015066 001002 BNE BBERR3 ;MEMORY PARITY ERROR OCCURRED.
3244 015070 000137 054076 JMP SPUR
3245
3246 015074 BBERR3: ;REPORT ERROR.
3247 015074 013737 015006 001634 MOV BBFLG1, $TMP3
3248 015102 012637 001632 MOV (SP)+, $TMP2
3249 015106 005726 TST (SP)+
3250 015110 013737 177744 001636 MOV @MEMERR, $TMP4
3251 015116 013737 177740 001644 MOV @LOADRS, $TMP7
3252 015124 013737 177742 001646 MOV @HIADRS, $TMP10
3253 015132 013737 014772 001640 MOV BBADR1, $TMP5
3254 015140 013737 014774 001642 MOV BBADR1+2, $TMP6
3255 015146 104040 1$: ERROR 40
3256
3257 015150 053737 014772 014776 BIS BBADR1, BBADR2 ;COMPUTE LOGICAL 'OR' OF
3258 015156 053737 014774 015000 BIS BBADR1+2, BBADR2+2 ;BAD ADDRESSES.
    
```

```

3259 015164 005137 015002          COM    BBADR3          ;COMPUT THE LOGICAL 'AND'
3260 015170 043737 014772 015002  BIC    BBADR1, BBADR3 ;OF THE BAD ADDRESSES.
3261 015176 005137 015002          COM    BBADR3
3262 015202 005137 015004          COM    BBADR3+2
3263 015206 043737 014774 015004  BIC    BBADR1+2, BBADR3+2
3264 015214 005137 015004          COM    BBADR3+2
3265
3266 015220 012737 177777 015010  MOV    #-1, BBFLG2    ;SET THE ERROR FLAG.
3267 015226 005237 015012          INC    BBCNT1        ;INCREMENT THE ERROR
3268 015232 005537 015014          ADC    BBCNT1+2      ;COUNT.
3269
3270 015236 012737 015260 000114  MOV    #BBERR4, @#CACHVEC ;TRY TO GET THE BAD
3271                                     ;ADDRESS OUT OF THE ADDRESS
3272                                     ;MEMORY.
3273 015244 013705 177740          MOV    @#LOADRS, R5
3274 015250 042705 176001          BIC    #176001, R5
3275 015254 005715                   TST    (R5)
3276 015256 000401                   BR     BBERR5
3277 015260 022626                   BBERR4: CMP    (SP)+, (SP)+
3278 015262 012737 177777 177744  BBERR5: MOV    #-1, @#MEMERR
3279 015270 012737 015022 000114  MOV    #BBERR1, @#CACHVEC
3280 015276 000137 014674          JMP    BB6
3281
3282 015302 104416                   BBDONE: RSET          ;DONE!
3283

```

```

3284 ;*****
3285 ;*TEST 15      CACHE ADDRESS MEMORY PARITY LOGIC TEST
3286 ;*
3287 ;*THIS IS A TEST OF THE PARITY CHECKERS AND PARITY GENERATOR
3288 ;*OF THE CACHE ADDRESS MEMORY. EVERY POSSIBLE ADDRESS TAG,
3289 ;*BITS 21 THROUGH 10, WHICH CAN BE STORED IN THE CACHE
3290 ;*ADDRESS MEMORY IS GENERATED, MADE A HIT AND THE
3291 ;*MAINTENANCE REGISTER IS THEN USED TO FORCE A CACHE ADDRESS
3292 ;*MEMORY PARITY ERROR AT EACH OF THE ADDRESSES
3293 ;*GENERATED. NOTE THAT BITS 9 THROUGH 0 OF THE ADDRESSES
3294 ;*IS NOT OF CONCERN, SO THESE BITS WILL BE THE SAME
3295 ;*FOR EACH ADDRESS; THIS IS BECAUSE ONLY BITS 21 THROUGH
3296 ;*10 ARE STORED IN THE ADDRESS MEMORY THEREFORE ONLY
3297 ;*THESE BITS ARE PARITY CHECKED IN THE CACHE ADDRESS
3298 ;*MEMORY PARITY CHECKERS. ALSO NOTE THAT THE RANGE
3299 ;*OF THE ADDRESSES MUST BE LIMITED TO BETWEEN THE
3300 ;*BOUNDS IMPOSED BY THE HIGHEST AVAILABLE MEMORY WORD
3301 ;*AND THE LAST WORD OF MEMORY USED BY THIS PROGRAM.
3302 ;*THE MANNER IN WHICH THE ERROR WILL BE FORCED
3303 ;*WILL BE TO PUT THE INSTRUCTIONS:
3304 ;*      1$:      MOV    R4, (R2)
3305 ;*      TSTADS:  CLR    (R2)
3306 ;*      RTS     PC
3307 ;*AT THE PARTICULAR ADDRESS BEING TESTED, WHERE
3308 ;*'TSTADS' IS THE ADDRESS BEING TESTED. R4 CONTAINS
3309 ;*A PATTERN TO BE LOADED IN THE MAINTENANCE REGISTER
3310 ;*WHICH WILL FORCE AN ERROR IN THE CACHE ADDRESS
3311 ;*MEMORY; R2 CONTAINS THE ADDRESS OF THE MAINTENANCE
3312 ;*REGISTER. NOTE FOR EACH ADDRESS R4 WILL FIRST
3313 ;*BE SUCH AS TO CAUSE AN ERROR IN THE LOW
3314 ;*BYTE ADDRESS PARITY CHECKER THEN AT THE SAME

```

```

3315      ;*ADDRESS AN ERROR WILL BE FORCED ON THE HIGH BYTE!
3316      ;*THE SEQUENCE OF TEST ADDRESSES WILL BE GENERATED
3317      ;*TWICE ONCE MAKING THEM HITS IN GROUP 0 THEN
3318      ;*MAKING THEM HITS IN GROUP 1.
3319      ;*
3320      ;*****
3321 015304 000004      TST15: SCOPE
3322 015306 012737 000002 001676      MOV      #2,$TIMES      ;;DO 2 ITERATIONS
3323      000015      AA=$TN-1
3324      ;
3325 015314 012737 016374 054230      MOV      #TST16,SKAD      ;SET THE SKAD REGISTER
3326      ;
3327 015322 113737 001502 001626      MOV      $TSTNM,$TMP0      ;IN CASE THE TEST ABORTS.
3328 015330 012737 054076 000114      MOV      #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
3329 015336 104422      MMSKIP
3330      ;
3331 015340 012700 172340      MOV      #KIPAR0,R0      ;INITIALLY PUT MEMORY
3332 015344 012701 077406      MOV      #77406,R1      ;MANAGEMENT IN A 'PASSIVE'
3333 015350 012702 172300      MOV      #KIPDR0,R2      ;STATE, THAT IS MAP ALL
3334 015354 012703 000010      MOV      #10,R3          ;VIRTUAL ADDRESSES ON TO
3335 015360 010122      64$: MOV      R1,(R2)+        ;THEMSELVES AS PHYSICAL
3336 015362 077302      SOB      R3,64$          ;ADDRESSES.
3337 015364 005020      CLR      (R0)+
3338 015366 012720 000200      MOV      #200,(R0)+
3339 015372 012720 000400      MOV      #400,(R0)+
3340 015376 012720 000600      MOV      #600,(R0)+
3341 015402 012720 001000      MOV      #1000,(R0)+
3342 015406 012720 001200      MOV      #1200,(R0)+
3343 015412 012720 001400      MOV      #1400,(R0)+
3344 015416 012710 177600      MOV      #177600,(R0)
3345      ;
3346 015422 104424      AALOAD: SIZE
3347 015424 000000      AAHIAD: .WORD 0          ;ADDRESS OF THE HIGHEST
3348 015426 000000      .WORD 0                ;WORD IN MEMORY.
3349 015430 042737 000002 015424      BIC      #2,AALOAD
3350      ;
3351 015436 012700 016234      MOV      #AATMP2,R0      ;ESTABLISH BITS 9 THROUGH
3352 015442 042700 176003      BIC      #176003,R0      ;0 TO BE PART OF ALL
3353 015446 010037 016220      MOV      R0,AAOFST        ;THE TEST ADDRESSES.
3354 015452 005037 016222      CLR      AAOFST+2
3355      ;
3356 015456 012737 000020 172516      MOV      #20,@MMR3      ;ENABLE 22-BIT MODE
3357 015464 012737 000001 177572      MOV      #1,@MMR0      ;ADDRESSING
3358      ;
3359 015472 012737 000030 016210      MOV      #SOM1,AAGS      ;TEST GROUP 0 FIRST, AAGS
3360 015500 005037 016204      CLR      AAFLG1          ;CONTAINS A PATTERN TO
3361 015504 012737 001400 016212      MOV      #1400,AAERGS    ;BE PUT IN THE CONTROL
3362 015512 012737 004420 016230      MOV      #4420,AAEXER    ;REGISTER. AAERGS CONTAINS
3363      ;
3364 015520 012737 000001 016216      AA1: MOV      #1,AAADR1+2  ;A PATTERN FOR THE MAINT. REG.
3365 015526 005037 016214      CLR      AAADR1          ;AAADR1 CONTAINS BITS
3366      ;
3367      ;
3368 015532 013737 016210 177746      MOV      AAGS,@#CONTRL   ;10 THROUGH 22 OF
3369      ;
3370      ;
    
```

```
3371 015540 AA2: ;GET THE TEST ADDRESS
3372 ;INTO THE AAADR2=AAADR1+AAOFST
3373 ;DOUBLE PRECISION ADDITION, UNSIGNED
3374 015540 013737 016214 016224 MOV AAADR1,AAADR2
3375 015546 013737 016216 016226 MOV AAADR1+2,AAADR2+2
3376 015554 063737 016220 016224 ADD AAOFST,AAADR2
3377 015562 005537 016226 ADC AAADR2+2
3378 015566 063737 016222 016226 ADD AAOFST+2,AAADR2+2
3379
3380
3381
3382 ;SEE IF THIS ADDRESS
3383 ;IS A REAL MEMORY LOCATION
3384 ;IF NOT THIS GROUP HAS
3385 ;BEEN TESTED.
3386
3387 ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3388 015574 023737 016226 015426 CMP AAADR2+2,AALOAD+2 ;COMPARE THE HIGH ORDER
3389 015602 001006 BNE 64$ ;PARTS OF AAADR2 AND ARG2.
3390 015604 023737 016224 015424 CMP AAADR2,AALOAD ;COMPARE THE LOW ORDER
3391
3392 015612 001002 BNE 64$ ;PARTS.
3393
3394
3395
3396 015614 000137 015632 JMP AA3 ;THEY WERE EQUAL!
3397
3398 015620 103402 64$: BLO 65$
3399 015622 000137 016142 JMP AA8 ;THE FIRST ADDRESS IS LARGER
3400 ;THAN THE SECOND!
3401 015626 000137 015632 65$: JMP AA3 ;THE FIRST IS LESS THAN THE
3402 ;SECOND.
3403
3404
3405 015632 012737 000001 016206 AA3: MOV #1,AAFLG2 ;THE ADDRESS IS GOOD! SET
3406 ;AAFLG2 TO INDICATE AN
3407 ;ERROR IS BEING FORCED
3408 ;ON THE LOW BYTE.
3409 ;ESTABLISH A VIRTUAL ADDRESS WHICH WILL RELOCATE
3410 ;THROUGH KIPAR6 TO THE TEST ADDRESS.
3411 015640 013703 016224 MOV AAADR2,R3
3412 015644 013702 016226 MOV AAADR2+2,R2
3413 015650 162703 000002 SUB #2,R3
3414 015654 005602 SBC R2
3415 015656 010300 MOV R3,R0
3416 015660 042700 177700 BIC #177700,R0 ;R0 CONTAINS THE VIRTUAL
3417 015664 062700 140000 ADD #140000,R0 ;ADDRESS.
3418
3419 015670 073227 177772 ASHC #-6,R2 ;SET KIPAR6
3420 015674 010337 172354 MOV R3,@#KIPAR6
3421
3422 015700 012737 054076 000114 MOV #SPUR,@#CACHVEC ;RESET VECTOR CACHVEC IN CASE
3423 ;A PARITY ERROR OCCURS
3424 ;WHILE SETTING UP THE
3425 ;INSTRUCTIONS AT THE TEST
3426 ;ADDRESS.
```

```

3427                                     ;PUT THE INSTRUCTIONS AT
3428                                     ;THE TEST ADDRESS
3429 015706 012710 010112                MOV    #010112,(R0)    ;010112 = 'MOV R4,(R2)'
3430 015712 012760 005012 000002        MOV    #005012,2(R0) ;005012 = 'CLR (R2)'
3431 015720 012760 000207 000004        MOV    #000207,4(R0) ;000207 = 'RTS PC'
3432
3433 015726 005760 000002                TST    2(R0)          ;MAKE THE TEST ADDRESS
3434 015732 005760 000002                TST    2(R0)          ;A HIT IN THE GROUP
3435 015736 032737 000010 177752        BIT    #10,@#HITMIS ;BEING TESTED!
3436 015744 001016                        BNE    AA4
3437
3438 015746 012737 015774 001634        MOV    #1$,$TMP3     ;IF UNABLE TO GET A GIT
3439 015754 013737 016204 001630        MOV    AAFLG1,$TMP1 ;REPORT ERROR!
3440 015762 010037 001632                MOV    R0,$TMP2
3441 015766 062737 000002 001632        ADD    #2,$TMP2
3442 015774 104001                        1$:    ERROR    1
3443 015776 000137 016124                JMP    AA7           ;GO TO NEXT TEST ADDRESS.
3444
3445 016002                                AA4:    ;THE TEST ADDRESS IS NOW
3446                                     ;A HIT IN THE GROUP
3447 016002 012737 016240 000114        MOV    #AAERR1,@#CACHVEC ;BEING TESTED. NOW RESET
3448                                     ;CACHVEC TO GO TO THE EXPECTED
3449                                     ;ERROR HANDLER
3450 016010 012702 177750                MOV    #MAINT,R2     ;SET R2
3451 016014 013704 016212                MOV    AAERGS,R4     ;SET R4 WHICH WILL BE
3452 016020 042704 005000                BIC    #5000,R4      ;LOADED INTO THE MAINT.
3453                                     ;REG SO AS TO FORCE
3454                                     ;A LOW BYTE ADDRESS
3455                                     ;MEMORY PARITY ERROR
3456                                     ;IN THE GROUP BEING
3457                                     ;TESTED.
3458 016024 000240                        NOP
3459 016026 004710                        JSR    PC,(R0)       ;FOR SCOPING WITH AN OSCILLOSCOPE.
3460                                     ;GO TO THE TEST
3461                                     ;ADDRESS!
3462 016030                                AA5:    ;RETURN,RTS PC, BACK TO
3463 016030 013737 016204 001632        MOV    AAFLG1,$TMP2 ;HERE IF THE TEST FAILED
3464 016036 013737 016224 001634        MOV    AAADR2,$TMP3 ;TO FORCE AN ERROR AT
3465 016044 013737 016226 001636        MOV    AAADR2+2,$TMP4 ;THE TEST ADDRESS'S LOW
3466 016052 104136                        1$:    ERROR    136    ;BYTE. REPORT THE FAILURE!
3467
3468 016054                                AA6:    ;TRY TO DO THE SAME
3469                                     ;THING NOW ONLY FORCE THE
3470                                     ;ERROR ON THE ADDRESSES
3471                                     ;HIGH BYTE!
3472                                     ;THE INSTRUCTIONS ARE
3473                                     ;ALREADY AT THE TEST
3474 016054 012737 000002 016206        MOV    #2,AAFLG2     ;ADDRESS. BUT MAKE SURE
3475 016062 005760 000002                TST    2(R0)          ;IT IS STILL A HIT!
3476 016066 013704 016212                MOV    AAERGS,R4     ;SET R4 TO FORCE THE
3477 016072 042704 002400                BIC    #2400,R4      ;ERROR ON THE HIGH BYTE.
3478 016076 004710                        JSR    PC,(R0)       ;GO DO THE TEST!
3479
3480 016100                                AA16:   ;RETURN,RTS PC, BACK TO HERE
3481 016100 013737 016204 001632        MOV    AAFLG1,$TMP2 ;IF THE TEST FAILED
3482 016106 013737 016224 001634        MOV    AAADR2,$TMP3 ;IN TRYING TO FORCE A
    
```

```

3483 016114 013737 016226 001636      MOV    AAADR2+2,$TMP4 ;ERROR ON THE HIGH BYTE
3484 016122 104137      1$:    ERROR    137    ;IN THE ADDRESS MEMORY
3485
3486 016124 062737 002000 016214 AA7:    ADD    #2000,AAADR1 ;INCREMENT BITS 21 THROUGH
3487 016132 005537 016216      ADC    AAADR1+2    ;10 OF THE TEST ADDRESS
3488 016136 000137 015540      JMP    AA2         ;AND GO TEST THIS NEW
3489                                ;TEST ADDRESS!
3490 016142 005737 016204      AA8:    TST    AAFLG1  ;SEE IF BOTH GROUPS HAVE
3491 016146 001111      BNE    AADONE     ;BEEN TESTED. IF NOT, GO
3492 016150 012737 004440 016230      MOV    #4440,AAEXER ;BACK TO AA1 TO TEST
3493 016156 012737 000044 016210      MOV    #S1M0,AAGS  ;GROUP ONE, OTHERWISE DONE.
3494 016164 012737 000001 016204      MOV    #1,AAFLG1
3495 016172 012737 006000 016212      MOV    #6000,AAERGS
3496 016200 000137 015520      JMP    AA1
3497
3498 016204 000000      AAFLG1: .WORD 0 ;A FLAG WHICH INDICATES
3499                                ;WHICH GROUP IS BEING TESTED
3500                                ;1 OR 0
3501 016206 000000      AAFLG2: .WORD 0 ;A FLAG WHICH INDICATES
3502                                ;WHETHER THE LOW BYTE (1)
3503                                ;THE HIGH BYTE (2) IS
3504                                ;BEING TESTED.
3505 016210 000000      AAGS:   .WORD 0 ;A PATTERN FOR THE CONTROL
3506                                ;REGISTER.
3507 016212 000000      AAERGS: .WORD 0 ;PATTERN FOR THE MAINT. REG.
3508 016214 000000      AAADR1: .WORD 0 ;BITS 21 THROUGH 10 OF
3509 016216 000000      .WORD 0 ;THE TEST ADDRESS.
3510 016220 000000      AAOFS:  .WORD 0 ;BITS 9 THROUGH 0 OF
3511 016222 000000      .WORD 0 ;THE TEST ADDRESS.
3512 016224 000000      AAADR2: .WORD 0 ;THE TEST ADDRESS
3513 016226 000000      .WORD 0 ;'AAADR2 = AAADR1+AAOFS'
3514 016230 000000      AAEXER: .WORD 0 ;EXPECTED ERROR REGISTER
3515 016232 000000      AATMP1: .WORD 0 ;THESE ADDRESSES ARE
3516 016234 000000      AATMP2: .WORD 0 ;USED TO DETERMINE AAOFS.
3517 016236 000000      .WORD 0
3518
3519 016240 016666 000002 000004 AAERR1: MOV    2(SP),4(SP) ;RESET THE STACK. RECALL THAT THE
3520 016246 012616      MOV    (SP)+,(SP) ;TEST ROUTINE WAS JSR'ED TO AND
3521                                ;A PARITY ERROR TRAP BROUGHT CONTROL
3522                                ;BACK!!
3523 016250 023737 016230 177744      CMP    AAEXER,@#MEMERR ;MAKE SURE THE ERROR
3524 016256 001405      BEQ    1$         ;WHICH OCCURRED WAS
3525 016260 012737 054076 000114      MOV    #SPUR,@#CACHVEC ;THE EXPECTED ERROR AT
3526 016266 000137 054076      JMP    SPUR       ;THE EXPECTED ADDRESS,
3527                                ;IF NOT GO TO THE
3528                                ;SPURIOUS ERROR HANDLER,
3529                                ;SPUR!
3530 016272      1$:
3531
3532                                ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3533 016272 023737 016226 177742      CMP    AAADR2+2,LOADRS+2 ;COMPARE THE HIGH ORDER
3534 016300 001006      BNE    64$       ;PARTS OF AAADR2 AND ARG2.
3535 016302 023737 016224 177740      CMP    AAADR2,LOADRS  ;COMPARE THE LOW ORDER
3536
3537 016310 001002      BNE    64$       ;PARTS.
3538
    
```

```

3539
3540
3541 016312 000137 016330          JMP    AAERR2          ;THEY WERE EQUAL!
3542
3543 016316 103402          64$:  BLO    65$
3544 016320 000137 .054076          JMP    SPUR           ;THE FIRST ADDRESS IS LARGER
3545                                     ;THAN THE SECOND!
3546 016324 000137 054076          65$:  JMP    SPUR           ;THE FIRST IS LESS THAN THE
3547                                     ;SECOND.
3548
3549
3550 016330 012737 177777 177744 AAERR2: MOV    #-1,@MEMERR ;IF EVERYTHING WAS
3551                                     ;CORRECT, CLR THE ERROR
3552 016336 022626          CMP    (SP)+,(SP)+   ;REGISTER RESET THE
3553                                     ;STACK AND CONTINUE
3554 016340 023727 016206 000002          CMP    AAFLG2,#2    ;TESTING
3555 016346 001002          BNE    1$
3556 016350 000137 016124          JMP    AA7           ;TEST THE NEXT ADDRESS
3557 016354 023727 016206 000001 1$:  CMP    AAFLG2,#1
3558 016362 001002          BNE    2$
3559 016364 000137 016054          JMP    AA6           ;TEST THE HIGH BYTE OF THIS ADDRESS
3560 016370 000000          2$:  HALT           ;???HOW DID WE GET HERE?
3561
3562 016372 104416          AADONE: RSET        ;DONE!
    
```


3563
 3564
 3565
 3566
 3567
 3568
 3569
 3570
 3571
 3572
 3573
 3574
 3575
 3576
 3577
 3578
 3579
 3580
 3581
 3582
 3583
 3584
 3585
 3586
 3587
 3588 016374 000004
 3589 016376 012737 000002 001676
 3590 000016
 3591 016404
 3592
 3593 016404 012737 020016 054230
 3594
 3595 016412 012737 054076 000114
 3596 016420 113737 001502 001626
 3597 016426 005037 017504
 3598 016432 104422
 3599
 3600 016434 104424
 3601 016436 000000
 3602 016440 000000
 3603
 3604 016442 005037 017500
 3605 016446 012737 000034 017522
 3606 016454 012737 000054 017524
 3607
 3608 016462 005037 017502
 3609 016466 012700 016404
 3610 016472 012701 001000
 3611
 3612 016476 013737 017522 177746
 3613 016504 005760 002000
 3614 016510 013737 017524 177746
 3615 016516 005720
 3616 016520 077112
 3617
 3618 016522 013701 017522

```

:*****
:*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
UUO:
        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        ;CLEAR THE PROGRESS FLAG.
        MOV      #UUO,R0      ;MAKE THIS CODE HITS, IN
        MOV      #1000,R1    ;THE GROUP NOT BEING TESTED.
UU2:   MOV      UUGS,@#CONTRL
        TST     2000(R0)
        MOV      UUGM,@#CONTRL
        TST     (R0)+
        SOB     R1,UU2
        MOV      UUGS,R1      ;SELECT THE GROUP BEING TESTED.

```

```

3619 016526 042701 177717      BIC      #177717,R1
3620 016532 010137 177746      MOV      R1,@#CONTRL
3621
3622
3623 016536 012700 172340      MOV      #KIPAR0,R0      ;INITIALLY PUT MEMORY
3624 016542 012701 077406      MOV      #77406,R1      ;MANAGEMENT IN A 'PASSIVE'
3625 016546 012702 172300      MOV      #KIPDR0,R2      ;STATE, THAT IS MAP ALL
3626 016552 012703 000010      MOV      #10,R3         ;VIRTUAL ADDRESSES ON TO
3627 016556 010122      64$: MOV      R1,(R2)+        ;THEMSELVES AS PHYSICAL
3628 016560 077302      SOB      R3,64$         ;ADDRESSES.
3629 016562 005020      CLR      (R0)+
3630 016564 012720 000200      MOV      #200,(R0)+
3631 016570 012720 000400      MOV      #400,(R0)+
3632 016574 012720 000600      MOV      #600,(R0)+
3633 016600 012720 001000      MOV      #1000,(R0)+
3634 016604 012720 001200      MOV      #1200,(R0)+
3635 016610 012720 001400      MOV      #1400,(R0)+
3636 016614 012710 177600      MOV      #177600,(R0)
3637
3638 016620 012737 000020 172516      MOV      #20,@MMR3      ;TURN ON MEMORY MANAGEMENT.
3639 016626 012737 000001 177572      MOV      #1,@MMR0
3640
3641 016634 005037 017512      CLR      UUADR2         ;INITIALIZE THE ADDRESSES.
3642 016640 005037 017514      CLR      UUADR2+2
3643 016644 012737 140000 017506      MOV      #140000,UUADR1
3644 016652 005037 017510      CLR      UUADR1+2
3645 016656 012701 000400      MOV      #400,R1        ;COUNTER.
3646 016662 012737 017530 000114      MOV      #UJERR1,@#CACHVEC
3647 016670 012737 000001 017502      MOV      #1,UJFLG2      ;KEEP TRACK OF TEST PROGRESS.
3648 016676
3649      UU3:
3650 016676 013737 017506 017516      ;DOUBLE PRECISION ADDITION, UNSIGNED
3651 016704 013737 017510 017520      MOV      UUADR1,UUADR3
3652 016712 063737 017512 017516      MOV      UUADR1+2,UUADR3+2
3653 016720 005537 017520      ADD      UUADR2,UUADR3
3654 016724 063737 017514 017520      ADC      UUADR3+2
3655      ADD      UUADR2+2,UUADR3+2
3656
3657
3658
3659 016732      UU4:
3660      ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3661      CMP      UUADR3+2,UJLOAD+2      ;COMPARE THE HIGH ORDER
3662 016732 023737 017520 016440      BNE      64$            ;PARTS OF UUADR3 AND ARG2.
3663 016740 001006      CMP      UUADR3,UJLOAD      ;COMPARE THE LOW ORDER
3664 016742 023737 017516 016436
3665      BNE      64$            ;PARTS.
3666 016750 001002
3667
3668
3669
3670 016752 000137 017004      JMP      UU6            ;THEY WERE EQUAL.
3671
3672 016756 103402      64$: BLO      65$
3673 016760 000137 016770      JMP      UU5            ;THE FIRST ADDRESS IS LARGER
3674      ;THAN THE SECOND.

```

H 8

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 73
 CEKBD.D.P11 16-MAY-79 08:58 T16 CACHE ADDRESS MEMORY DUAL ADDRESS TEST, UPWARD SEQ 009C

```

3675 016764 000137 017004      65$:  JMP      UU6          ;THE FIRST IS LESS THAN THE
3676                                     ;SECOND.
3677
3678
3679 016770 012737 140000 017506 UU5:  MOV      #140000,UUADR1 ;RESET TO GET VALID ADDRESS.
3680 016776 005037 017510      CLR      UUADR1+2
3681 017002 000735      BR       UU3
3682
3683 017004 012702 017516      UU6:  MOV      #UUADR3,R2
3684
3685 017010 011203      MOV      (R2),R3          ;GET THE PHYSICAL ADDRESS POINTED
3686 017012 042703 177700      BIC      #177700,R3      ;TO BY R2 AND ESTABLISH
3687 017016 011205      MOV      (R2),R5 ;A VIRTUAL ADDRESS WHICH
3688 017020 016204 000002      MOV      2(R2),R4        ;WILL RELOCATE THROUGH
3689 017024 073427 177772      ASHC     #-6,R4          ;KIPAR6. SETUP KIPAR6 AND
3690 017030 010537 172354      MOV      R5,@#KIPAR6    ;LEAVE THE VIRTUAL ADDRESS
3691 017034 062703 140000      ADD      #140000,R3      ;IN R3.
3692
3693
3694 017040 005713      TST      (R3)            ;GET A HIT AT THE TEST
3695 017042 005713      TST      (R3)            ;ADDRESS.
3696
3697 017044 032737 000010 177752      BIT      #10,@#HITMIS
3698 017052 001012      BNE      UU7
3699 017054 013737 017500 001632      MOV      UUFLG1,$TMP2
3700 017062 013737 017516 001634      MOV      UUADR3,$TMP3
3701 017070 013737 017520 001636      MOV      UUADR3+2,$TMP4
3702 017076 104041      1$:  ERROR  41
3703 017100 062737 002000 017506 UU7:  ADD      #2000,UUADR1
3704 017106 005537 017510      ADC      UUADR1+2
3705 017112 062737 000004 017512      ADD      #4,UUADR2      ;LOOP TO WRITE NEXT ADDRESS
3706 017120 005301      DEC      R1
3707 017122 001402      BEQ     1$
3708 017124 000137 016676      JMP     JU3
3709 017130 012737 000002 017502 1$:  MOV      #2,UUFLG2
3710
3711 017136 013700 017524      MOV      UUGM,R0          ;FROM NOW ON SELECT THE
3712 017142 042700 177717      BIC      #177717,R0      ;GROUP NOT BEING TESTED.
3713 017146 010037 177746      MOV      R0,@#CONTRL
3714
3715 017152 005037 017512      UU8:  CLR      UUADR2          ;NOW RE-GENERATE ALL THE
3716 017156 005037 017514      CLR      UUADR2+2        ;ADDRESS WHICH WERE MADE
3717 017162 012737 140000 017506      MOV      #140000,UUADR1 ;HITS, ABOVE, AND MAKE SURE
3718 017170 005037 017510      CLR      UUADR1+2        ;THEY ARE STILL HITS.
3719 017174 012701 000400      MOV      #400,R1
3720 017200 012737 000003 017502      MOV      #3,UUFLG2
3721 017206
3722      UU9:  ;DOUBLE PRECISION ADDITION, UNSIGNED
3723 017206 013737 017506 017516      MOV      UUADR1,UUADR3
3724 017214 013737 017510 017520      MOV      UUADR1+2,UUADR3+2
3725 017222 063737 017512 017516      ADD      UUADR2,UUADR3
3726 017230 005537 017520      ADC      UUADR3+2
3727 017234 063737 017514 017520      ADD      UUADR2+2,UUADR3+2
3728
3729
3730

```

```

3731
3732 017242          UU10:
3733
3734          ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3735 017242 023737 017520 016440  CMP      UUADR3+2,UULOAD+2      ;COMPARE THE HIGH ORDER
3736 017250 001006          BNE      64$                    ;PARTS OF UUADR3 AND ARG2.
3737 017252 023737 017516 016436  CMP      UUADR3,UULOAD          ;COMPARE THE LOW ORDER
3738
3739 017260 001002          BNE      64$                    ;PARTS.
3740
3741
3742
3743 017262 000137 017314          JMP      UU12                    ;THEY WERE EQUAL!
3744
3745 017266 103402          64$:   BLO      65$
3746 017270 000137 017300          JMP      UU11                    ;THE FIRST ADDRESS IS LARGER
3747          ;THAN THE SECOND!
3748 017274 000137 017314          65$:   JMP      UU12                    ;THE FIRST IS LESS THAN THE
3749          ;SECOND.
3750
3751
3752 017300 012737 140000 017506  UU11:   MOV      #140000,UUADR1        ;RESFT TO GET A VALID ADDRESS.
3753 017306 005037 017510          CLR      UUADR1+2
3754 017312 000735          BR       UU9
3755
3756 017314 012702 017516          UU12:   MOV      #UUADR3,R2
3757
3758 017320 011203          MOV      (R2),R3                ;GET THE PHYSICAL ADDRESS POINTED
3759 017322 042703 177700          BIC      #177700,R3            ;TO BY R2 AND ESTABLISH
3760 017326 011205          MOV      (R2),R5 ;A VIRTUAL ADDRESS WHICH
3761 017330 016204 000002          MOV      2(R2),R4              ;WILL RELOCATE THROUGH
3762 017334 073427 177772          ASHC    #-6,R4                ;KIPAR6. SETUP KIPAR6 AND
3763 017340 010537 172354          MOV      R5,@KIPAR6           ;LEAVE THE VIRTUAL ADDRESS
3764 017344 062703 140000          ADD      #140000,R3           ;IN R3.
3765
3766
3767 017350 005713          TST      (R3) ;STILL A HIT?
3768 017352 032737 000010 177752  BIT      #10,@WHITMIS
3769 017360 001012          BNE      UU13
3770          ;NOT A HIT, A DUAL ADDRESSING
3771 017362 013737 017500 001632  MOV      UUFLG1,$TMP2          ;PROBLEM?
3772 017370 013737 017516 001634  MOV      UUADR3,$TMP3
3773 017376 013737 017520 001636  MOV      UUADR3+2,$TMP4
3774 017404 104042          1$:   ERROR      42
3775
3776 017406 062737 002000 017506  UU13:   ADD      #2000,UUADR1
3777 017414 005537 017510          ADC      UUADR1+2
3778 017420 062737 000004 017512  ADD      #4,UUADR2              ;LOOP TO READ NEXT ADDRESS
3779 017426 005301          DEC      R1
3780 017430 001402          BEQ     1$
3781 017432 000137 017206          JMP     UU9
3782 017436 012737 000004 017502  1$:   MOV      #4,UUFLG2
3783 017444 005737 017500  UU14:   TST      UUFLG1 ;TESTED BOTH GROUPS?
3784 017450 001161          BNE     UUDONE                ;YES.
3785 017452 012737 000001 017500  MOV      #1,UUFLG1            ;NO, GO TEST GROUP 1.
3786 017460 012737 000054 017522  MOV      #S1MOM1,UUGS
    
```

```

3787 017466 012737 000034 017534      MOV    #SOMOM1,UUGI
3788 017474 000137 016462              JMP    UU1
3789
3790 017500 000000              JUFLG1: .WORD 0      ;FLAG WHICH DESIGNATES
3791                                ;WHICH GROUP IS BEING TESTED,
3792                                ;1 OR 0.
3793 017502 000000              UUFLG2: .WORD 0      ;DESIGNATES HOW FAR THE
3794                                ;TEST HAS PROGRESSED.
3795 017504 000000              UUFLG3: .WORD 0      ;ERROR DURING TEST UUFLG2-4
3796                                ;PHASE.
3797 017506 000000              UUADR1: .WORD 0      ;ADDRESS WRITTEN INTO ADDRESS
3798 017510 000000              .WORD 0              ;MEMORY LOCATION
3799 017512 000000              UUADR2: .WORD 0      ;ADDRESS MEMORY LOCATION
3800 017514 000000              .WORD 0              ;BEING TESTED
3801 017516 000000              UUADR3: .WORD 0      ;TEST ADDRESS:UUADR3=UUADR1+UUADR2
3802 017520 000000              .WORD 0
3803
3804 017522 000000              UUGS:   .WORD 0      ;PATTERNS FOR THE CACHE CONTROL
3805 017524 000000              UUGM:   .WORD 0      ;REGISTER.
3806 017526 000000              UUTMP:  .WORD 0
3807
3808 017530 032737 000060 177744      UUERR1: BIT    #60,@MEMERR ;WAS THE ERROR A CACHE ADDRESS
3809 017536 001002              BNE     UUERR2        ;MEMORY PARITY ERROR
3810 017540 000137 054076              JMP     SPUR
3811
3812 017544              UUERR2:              ;REPORT ERROR.
3813 017544 012637 001632              MOV    (SP)+,$TMP2
3814 017550 005726              TST   (SP)+
3815 017552 013737 017500 001634      MOV    UUFLG1,$TMP3
3816 017560 013737 177744 001636      MOV    @MEMERR,$TMP4
3817 017566 013737 017516 001640      MOV    UUADR3,$TMP5
3818 017574 013737 017520 001642      MOV    UUADR3+2,$TMP6
3819 017602 013737 177740 001644      MOV    @LOADRS,$TMP7
3820 017610 013737 177742 001646      MOV    @HIADRS,$TMP10
3821 017616 104043              1$:   ERROR 43
3822
3823 017620 042737 177717 001636      BIC   #177717,$TMP4 ;TRY TO GET THE BAD ADDRESS
3824 017626 013737 177746 017526      MOV    @CONTRL,UUTMP ;OUT OF THE ADDRESS MEMORY.
3825 017634 012737 017664 000114      MOV    #UUERR3,@CACHVEC
3826 017642 013705 177740              MOV    @LOADRS,R5
3827 017646 042705 176001              BIC   #176001,R5
3828 017652 013737 001636 177746      MOV    $TMP4,@CONTRL
3829 017660 005715              TST   (R5)
3830 017662 000401              BR    UUERR4
3831 017664 022626              UUERR3: CMP   (SP)+,(SP)+
3832 017666 012737 177777 177744      UUERR4: MOV   #-1,@MEMERR
3833
3834 017674 013737 017526 177746      MOV    UUTMP,@CONTRL ;RESET THE CONTROL REGISTER.
3835 017702 012737 017530 000114      MOV    #UUERR1,@CACHVEC
3836
3837 017710 023727 017502 000001              CMP   UUFLG2,#1      ;RETURN, USING UUFLG2 TO
3838 017716 001002              BNE   1$             ;DECIDE WHERE.
3839 017720 000137 017100              JMP   UU7
3840 017724 023727 017502 000002 1$:   CMP   UUFLG2,#2
3841 017732 001002              BNE   2$
3842 017734 000137 017152              JMP   UU8

```

```

3843 017740 023727 017502 000003 2$:  CMP      UUFLG2,#3
3844 017746 001002                BNE      3$
3845 017750 000137 017406                JMP      UU13
3846 017754 023727 017502 000004 3$:  CMP      UUFLG2,#4
3847 017762 001007                BNE      4$
3848 017764 005737 017504                TST      UUFLG3
3849 017770 001011                BNE      UUDONE
3850 017772 005337 017504                DEC      UUFLG3
3851 017776 000137 017444                JMP      UU14
3852
3853 020002 005737 017502                4$:  TST      UUFLG2
3854 020006 001002                BNE      UUDONE          ;??HALT???
3855 020010 000137 016462                JMP      UU1
3856
3857 020014 104416                UUDONE:RSET             ;DONE.
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883

```

```

:*****
:*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.
:*
:*****

```

```

3884 020016 000004                TST17: SCOPE
3885 020020 012737 000002 001676      MOV      #2,$TIMES      ;;DO 2 ITERATIONS
3886 020026 000017                VV=$TN-1
3887 020026                VV0:
3888                                ;SET THE SKAD REGISTER
3889 020026 012737 021444 054230      MOV      #TST20,SKAD   ;IN CASE THE TEST ABORTS.
3890
3891 020034 012737 054076 000114      MOV      #SPUR,@#CACHVEC ;INITIALLY EXPECT NO ERRORS.
3892 020042 113737 001502 001626      MOV      $TSTM,$TMP0
3893
3894 020050 005037 021132                CLR      VVFLG3        ;CLEAR THE ERROR FLAG.
3895
3896 020054 104422                MMSKIP
3897
3898 020056 104424                SIZE

```

```

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) L 8 16-MAY-79 09:11 PAGE 77
CEKBDD.P11 16-MAY-79 08:58 T17 CACHE ADDRESS MEMORY DIJAL ADDRESS TEST, DOWNWARD SEQ 0102

3899 020060 000000 VVLOAD: .WORD 0 ;ADDRESS OF THE HIGHEST
3900 020062 000000 VVHIAD: .WORD 0 ;WORD IN MEMORY.
3901
3902 020064 005037 021126 CLR VVFLG1 ;TEST GROUP 0 FIRST
3903 020070 012737 000034 021150 MOV #SOMOM1,VVGS
3904 020076 012737 000054 021152 MOV #S1MOM1,VVGM
3905
3906 020104 005037 021130 VV1: CLR VVFLG2 ;CLEAR THE PROGRESS FLAG
3907 020110 012700 020026 MOV #VV0,R0 ;MAKE THIS CODE HITS IN
3908 020114 012701 001000 MOV #1000,R1 ;THE GROUP NOT BEING
3909 ;TESTED.
3910 020120 013737 021150 177746 VV2: MOV VVGS,@#CONTRL
3911 020126 005760 002000 TST 2000(R0)
3912 020132 013737 021152 177746 MOV VVGM,@#CONTRL
3913 020140 005720 TST (R0)+
3914 020142 077112 SOB R1,VV2
3915
3916 020144 013700 021150 MOV VVGS,R0 ;FROM NOW ON SELECT
3917 020150 042700 177717 BIC #177717,R0 ;THE GROUP BEING TESTED.
3918 020154 010037 177746 MOV R0,@#CONTRL
3919
3920
3921 020160 012700 172340 MOV #KIPAR0,R0 ;INITIALLY PUT MEMORY
3922 020164 012701 077406 MOV #77406,R1 ;MANAGEMENT IN A 'PASSIVE'
3923 020170 012702 172300 MOV #KIPDR0,R2 ;STATE, THAT IS MAP ALL
3924 020174 012703 000010 MOV #10,R3 ;VIRTUAL ADDRESSES ON TO
3925 020200 010122 64$: MOV R1,(R2)+ ;THEMSELVES AS PHYSICAL
3926 020202 077302 SOB R3,64$ ;ADDRESSES.
3927 020204 005020 CLR (R0)+
3928 020206 012720 000200 MOV #200,(R0)+
3929 020212 012720 000400 MOV #400,(R0)+
3930 020216 012720 000600 MOV #600,(R0)+
3931 020222 012720 001000 MOV #1000,(R0)+
3932 020226 012720 001200 MOV #1200,(R0)+
3933 020232 012720 001400 MOV #1400,(R0)+
3934 020236 012710 177600 MOV #177600,(R0)
3935
3936 020242 012737 000020 172516 MOV #20,@#MMR3 ;TURN ON MEMORY MANAGEMENT.
3937 020250 012737 000001 177572 MOV #1,@#MMR0
3938
3939 020256 012737 001774 021140 MOV #1774,VVADR2 ;INITIALIZE THE ADDRESSES
3940 020264 005037 021142 CLR VVADR2+2
3941 020270 012737 140000 021134 MOV #140000,VVADR1
3942 020276 005037 021136 CLR VVADR1+2
3943 020302 012701 000400 MOV #400,R1 ;A COUNTER.
3944 020306 012737 021156 000114 MOV #VVERR1,@#CACHVEC ;EXPECT ERRORS NOW.
3945 020314 012737 000001 021130 MOV #1,VVFLG2 ;KEEP TRACK OF TEST PROGRESS.
3946
3947 020322 VV3:
3948 ;DOUBLE PRECISION ADDITION, UNSIGNED
3949 020322 013737 021134 021144 MOV VVADR1,VVADR3
3950 020330 013737 021136 021146 MOV VVADR1+2,VVADR3+2
3951 020336 063737 021140 021144 ADD VVADR2,VVADR3
3952 020344 005537 021146 ADC VVADR3+2
3953 020350 063737 021142 021146 ADD VVADR2+2,VVADR3+2
3954

```

```

3955
3956
3957
3958 020356          VV4:
3959
3960          ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
3961 020356 023737 021146 020062  CMP    VVADR3+2,VVLOAD+2      ;COMPARE THE HIGH ORDER
3962 020364 001006          BNE    64$                   ;PARTS OF VVADR3 AND ARG2.
3963 020366 023737 021144 020060  CMP    VVADR3,VVLOAD        ;COMPARE THE LOW ORDER
3964
3965 020374 001002          BNE    64$                   ;PARTS.
3966
3967
3968
3969 020376 000137 020430          JMP    VV6                   ;THEY WERE EQUAL!
3970
3971 020402 103402          64$:  BLO    65$
3972 020404 000137 020414          JMP    VV5                   ;THE FIRST ADDRESS IS LARGER
3973                                     ;THAN THE SECOND!
3974 020410 000137 020430          65$:  JMP    VV6                   ;THE FIRST IS LESS THAN THE
3975                                     ;SECOND.
3976
3977
3978 020414 012737 140000 021134  VV5:  MOV    #140000,VVADR1      ;RESET TO GET A VALID ADDRESS.
3979 020422 005037 021136          CLR    VVADR1+2
3980 020426 000735          BR     VV3
3981
3982 020430 012702 021144          VV6:  MOV    #VVADR3,R2
3983
3984 020434 011203          MOV    (R2),R3              ;GET THE PHYSICAL ADDRESS POINTED
3985 020436 042703 177700          BIC    #177700,R3          ;TO BY R2 AND ESTABLISH
3986 020442 011205          MOV    (R2),R5              ;A VIRTUAL ADDRESS WHICH
3987 020444 016204 000002          MOV    2(R2),R4            ;WILL RELOCATE THROUGH
3988 020450 073427 177772          ASHC   #-6,R4              ;KIPAR6. SETUP KIPAR6 AND
3989 020454 010537 172354          MOV    R5,#KIPAR6          ;LEAVE THE VIRTUAL ADDRESS
3990 020460 062703 140000          ADD    #140000,R3          ;IN R3.
3991
3992
3993 020464 005713          TST    (R3)                 ;GET A HIT AT THE
3994 020466 005713          TST    (R3)                 ;TEST ADDRESS
3995 020470 032737 000010 177752  BIT    #10,#HITMIS
3996 020476 001012          BNE    VV7
3997                                     ;REPORT FAILURE TO GET A HIT.
3998 020500 013737 021126 001632  MOV    VVFLG1,$TMP2
3999 020506 013737 021144 001634  MOV    VVADR3,$TMP3
4000 020514 013737 021146 001636  MOV    VVADR3+2,$TMP4
4001 020522 104041          1$:  ERROR 41
4002
4003 020524 062737 002000 021134  VV7:  ADD    #2000,VVADR1
4004 020532 005537 021136          ADC    VVADR1+2
4005 020536 062737 177774 021140  ADD    #-4,VVADR2          ;LOOP TO WRITE NEXT ADDRESS
4006 020544 005301          DEC    R1
4007 020546 001402          BEQ   1$
4008 020550 000137 020322          JMP   VV3
4009 020554 012737 000002 021130  1$:  MOV    #2,VVFLG2
4010
    
```



```

4011 020562 013700 021152      MOV      VVGM,R0      ;FROM NOW ON SELECT
4012 020566 042700 177717      BIC      #177717,R0   ;THE GROUP NOT BEING
4013 020572 010037 177746      MOV      R0,#CONTRL  ;TESTED.
4014
4015 020576 012737 001774 021140 VV8:   MOV      #1774,VVADR2 ;NOW RE-GENERATE ALL THE
4016 020604 005037 021142      CLR      VVADR2+2    ;ADDRESSES MADE HITS IN
4017 020610 012737 140000 021134      MOV      #140000,VVADR1 ;THE ABOVE PORTION OF
4018 020616 005037 021136      CLR      VVADR1+2    ;THE TEST, AND MAKE SURE
4019 020622 012701 000400      MOV      #400,R1     ;THEY ARE STILL HITS.
4020 020626 012737 000003 021130      MOV      #3,VVFLG2
4021 020634
4022
4023 020634 013737 021134 021144      ;DOUBLE PRECISION ADDITION, UNSIGNED
4024 020642 013737 021136 021146      MOV      VVADR1,VVADR3
4025 020650 063737 021140 021144      MOV      VVADR1+2,VVADR3+2
4026 020656 005537 021146      ADD      VVADR2,VVADR3
4027 020662 063737 021142 021146      ADC      VVADR3+2
4028
4029
4030
4031
4032 020670
4033
4034
4035 020670 023737 021146 020062      ;DOUBLE PRECISION COMPARE OF TWO 22-BIT ADDRESSES
4036 020676 001006
4037 020700 023737 021144 020060      CMP      VVADR3+2,VVLOAD+2 ;COMPARE THE HIGH ORDER
4038
4039 020706 001002
4040
4041
4042
4043 020710 000137 020742      BNE      64$         ;PARTS OF VVADR3 AND ARG2.
4044
4045 020714 103402
4046 020716 000137 020726      BNE      64$         ;COMPARE THE LOW ORDER
4047
4048 020722 000137 020742      BNE      64$         ;PARTS.
4049
4050
4051
4052 020726 012737 140000 021134 VV11:  MOV      #140000,VVADR1 ;RESET TO CREATE A VALID
4053 020734 005037 021136      CLR      VVADR1+2    ;ADDRESS
4054 020740 000735
4055
4056 020742 012702 021144      BR       VV9
4057
4058 020746 011203
4059 020750 042703 177700      MOV      #VVADR3,R2
4060 020754 011205
4061 020756 016204 000002      MOV      (R2),R3     ;GET THE PHYSICAL ADDRESS POINTED
4062 020762 073427 177772      BIC      #177700,R3  ;TO BY R2 AND ESTABLISH
4063 020766 010537 172354      MOV      (R2),R5 ;A VIRTUAL ADDRESS WHICH
4064 020772 062703 140000      MOV      2(R2),R4    ;WILL RELOCATE THROUGH
4065
4066
4067
4068
4069
4070
4071
4072
4073
4074
4075
4076
4077
4078
4079
4080
4081
4082
4083
4084
4085
4086
4087
4088
4089
4090
4091
4092
4093
4094
4095
4096
4097
4098
4099
4100
    
```

```

4067 020776 005713          TST      (R3)          ;STILL A HIT?
4068 021000 032737 000010 177752 BIT      #10,@#HITMIS
4069 021006 001012          BNE     VV13          ;REPORT ERROR.
4070
4071 021010 013737 021126 001632 MOV     VVFLG1,$TMP2
4072 021016 013737 021144 001634 MOV     VVADR3,$TMP3
4073 021024 013737 021146 001636 MOV     VVADR3+2,$TMP4
4074 021032 104042          1$:     ERROR      42
4075
4076 021034 062737 002000 021134 VV13:   ADD     #2000,VVADR1
4077 021042 005537 021136          ADC     VVADR1+2
4078 021046 062737 177774 021140          ADD     #-4,VVADR2
4079 021054 005301          DEC     R1
4080 021056 001402          BEQ     1$
4081 021060 000137 020634          JMP     VV9
4082 021064 012737 000004 021130 1$:     MOV     #4,VVFLG2
4083 021072 005737 021126          VV14:   TST     VVFLG1          ;TESTED BOTH GROUPS?
4084 021076 001161          BNE     VVDONE        ;YES.
4085 021100 012737 000034 021152          MOV     #S1MOM1,VVGM  ;NO GO TEST GROUP 1.
4086 021106 012737 000054 021150          MOV     #S1MOM1,VVGS
4087 021114 012737 000001 021126          MOV     #1,VVFLG1
4088 021122 000137 020104          JMP     VV1
4089
4090 021126 000000          VVFLG1: .WORD 0          ;0 OR 1, GROUP BEING TESTED.
4091 021130 000000          VVFLG2: .WORD 0          ;TEST PROGRESS FLAG.
4092 021132 000000          VVFLG3: .WORD 0          ;ERROR FLAG.
4093
4094 021134 000000          VVADR1: .WORD 0          ;PATTERN WRITTEN INTO THE ADDRESS
4095 021136 000000          .WORD 0          ;MEMORY LOCATION.
4096 021140 000000          VVADR2: .WORD 0          ;ADDRESS MEMORY LOCATION BEING
4097 021142 000000          .WORD 0          ;TESTED X 4.
4098 021144 000000          VVADR3: .WORD 0          ;TEST ADDRESS.
4099 021146 000000          .WORD 0          ;VVADR3=VVADR2+VVADR1
4100
4101 021150 000000          VVGS:   .WORD 0          ;PATTERNS FOR THE CACHE
4102 021152 000000          VVGM:   .WORD 0          ;CONTROL REGISTER.
4103
4104 021154 000000          VVTMP:  .WORD 0
4105
4106 021156 032737 000060 177744 VVERR1: BIT     #60,@#MEMERR ;WAS THE ERROR THAT CAUSED
4107 021164 001002          BNE     VVERR2        ;THE TRAP TO HERE A CACHE
4108 021166 000137 054076          JMP     SPUR          ;ADDRESS MEMORY PARITY ERROR?
4109
4110          VVERR2:          ;REPORT ERROR.
4111 021172 012637 001632          MOV     (SP)+,$TMP2
4112 021176 005726          TST     (SP)+
4113 021200 013737 021126 001634          MOV     VVFLG1,$TMP3
4114 021206 013737 177744 001636          MOV     @#MEMERR,$TMP4
4115 021214 013737 021144 001640          MOV     VVADR3,$TMP5
4116 021222 013737 021146 001642          MOV     VVADR3+2,$TMP6
4117 021230 013737 177740 001644          MOV     @#LOADRS,$TMP7
4118 021236 013737 177742 001646          MOV     @#HIADRS,$TMP10
4119 021244 104043          1$:     ERROR      43
4120
4121 021246 042737 177717 001636          BIC     #177717,$TMP4 ;TRY TO GET THE BAD ADDRESS
4122 021254 013737 177746 021154          MOV     @#CONTRL,VVTMP ;OUT OF THE ADDRESS MEMORY.
    
```

```

4123 021262 012737 021312 000114      MOV      #VVERR3,@#CACHVEC
4124 021270 013705 177740      MOV      @#LOADRS,R5
4125 021274 042705 176001      BIC      #176001,R5
4126 021300 013737 001636 177746      MOV      $TMP4,@#CONTRL
4127 021306 005715      TST      (R5)
4128 021310 000401      BR       VVERR4
4129 021312 022626      VVERR3: CMP      (SP)+,(SP)+
4130 021314 012737 177777 177744      VVERR4: MOV      #-1,@#MEMERR
4131
4132 021322 013737 021154 177746      MOV      VVTMP,@#CONTRL ;RESET THE CONTRL REGISTER
4133 021330 012737 021156 000114      MOV      #VVERR1,@#CACHVEC
4134 021336 023727 021130 000001      CMP      VVFLG2,#1      ;RETURN, USING VVFLG2 TO
4135 021344 001002      BNE      1$              ;DECIDE WHERE.
4136 021346 000137 020524      JMP      VV7
4137 021352 023727 021130 000002 1$:      CMP      VVFLG2,#2
4138 021360 001002      BNE      2$
4139 021362 000137 020576      JMP      VV8
4140 021366 023727 021130 000003 2$:      CMP      VVFLG2,#3
4141 021374 001002      BNE      3$
4142 021376 000137 021034      JMP      VV13
4143 021402 023727 021130 000004 3$:      CMP      VVFLG2,#4
4144 021410 001007      BNE      4$
4145 021412 005737 021132      TST      VVFLG3
4146 021416 001011      BNE      VVDONE
4147 021420 005337 021132      DEC      VVFLG3
4148 021424 000137 021072      JMP      VV14
4149 021430 005737 021130      4$:      TST      VVFLG2
4150 021434 001002      BNE      VVDONE          ;????HALT???
4151 021436 000137 020104      JMP      VV1
4152
4153 021442 104416      VVDONE: RSE*           ;DONE.
4154
4155
4156

```

```

:*****
:*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.
:*
:*****

```

```

4168 021444 000004      TST20: SCOPE
4169 021446 012737 000010 001676      MOV      #10,$TIMES      ;;DO 10 ITERATIONS
4170      000020      CC=$TN-1
4171
4172 021454 012737 022230 054230      MOV      #TST21,SKAD      ;SET THE SKAD REGISTER
4173      ;IN CASE THE TEST ABORTS.
4174 021462 113737 001502 001626      MOV#B   $TSTNM,$TMP0
4175 021470 012737 021734 000114      MOV      #CCERR1,@#CACHVEC
4176
4177 021476 012737 000014 177746      MOV      #MOM1,@#CONTRL ;FORCE MISSES
4178

```

D 9

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 82
 CEKBDD.P11 16-MAY-79 08:58 T20 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ONES TEST SEQ 0107

```

4179 021504 012700 021730      MOV      #CCTMP2,R0      ;LOCATE THE TEST SPACE.
4180 021510 042700 000003      BIC      #3,R0
4181 021514 010001      MOV      R0,R1
4182 021516 005010      CC1:    CLR      (R0)      ;TEST MASK 0
4183 021520 005060 000002      CLR      2(R0)
4184 021524 000240      NOP
4185 021526 112711 000377      MOVB     #377,(R1)      ;FOR SCOPING WITH AN OSCILLOSCOPE.
4186 021532 022710 000377      CMP      #377,(R0)
4187 021536 001403      BEQ      CC3
4188 021540 004737 022146      CC2:    JSR      PC,CCERR3
4189 021544 000403      BR       CC4
4190 021546 005760 000002      CC3:    TST      2(R0)
4191 021552 001372      BNE      CC2
4192 021554 062701 000001      CC4:    ADD      #1,R1      ;TEST MASK 1.
4193 021560 005010      CLR      (R0)
4194 021562 005060 000002      CLR      2(R0)
4195 021566 000240      NOP
4196 021570 112711 000377      MOVB     #377,(R1)      ;FOR SCOPING WITH AN OSCILLOSCOPE.
4197 021574 022710 177400      CMP      #177400,(R0)
4198 021600 001403      BEQ      CC6
4199 021602 004737 022146      CC5:    JSR      PC,CCERR3
4200 021606 000403      BR       CC7
4201 021610 005760 000002      CC6:    TST      2(R0)
4202 021614 001372      BNE      CC5
4203
4204 021616 062701 000001      CC7:    ADD      #1,R1      ;TEST MASK 2.
4205 021622 005010      CLR      (R0)
4206 021624 005060 000002      CLR      2(R0)
4207 021630 000240      NOP
4208 021632 112711 000377      MOVB     #377,(R1)      ;FOR SCOPING WITH AN OSCILLOSCOPE.
4209 021636 022760 000377 000002      CMP      #377,2(R0)
4210 021644 001403      BEQ      CC9
4211 021646 004737 022146      CC8:    JSR      PC,CCERR3
4212 021652 000402      BR       CC10
4213 021654 005710      CC9:    TST      (R0)
4214 021656 001373      BNE      CC8
4215
4216 021660 062701 000001      CC10:   ADD      #1,R1      ;TEST MASK 3.
4217 021664 005010      CLR      (R0)
4218 021666 005060 000002      CLR      2(R0)
4219 021672 000240      NOP
4220 021674 112711 000377      MOVB     #377,(R1)      ;FOR SCOPING WITH AN OSCILLOSCOPE.
4221 021700 022760 177400 000002      CMP      #177400,2(R0)
4222 021706 001403      BEQ      CC12
4223 021710 004737 022146      CC11:   JSR      PC,CCERR3
4224 021714 000402      BR       CC13
4225 021716 005710      CC12:   TST      (R0)
4226 021720 001373      BNE      CC11
4227
4228 021722 000137 022226      CC13:   JMP      CCDONE
4229
4230 021726 000000      CCTMP1: .WORD    0
4231 021730 000000      CCTMP2: .WORD    0      ;THE TEST AREA.
4232 021732 000000      .WORD    0
4233
4234

```

E 9

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 83
 CEKBD.D.P11 16-MAY-79 08:58 T20 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ONES TEST SEQ 0108

```

4235 021734 032737 000002 177744 CCERR1: BIT #2,@MMEMERR ;SHOULD BE A MAIN MEMORY
4236 021742 001002 BNE 1$ ;ADDRESS AND CONTROL LINE
4237 021744 000137 054076 JMP SPUR ;PARITY ERROR.
4238 021750 020137 177740 1$: CMP R1,@WLOADRS ;ERROR ADDRESS SHOULD BE
4239 021754 001402 BEQ CCERR2 ;TEST ADDRESS.
4240 021756 000137 054076 JMP SPUR
4241 021762 012637 001642 CCERR2: MOV (SP)+,$TMP6
4242 021766 005037 001664 CLR $TMP17
4243 021772 005726 TST (SP)+ ;RESET THE STACK
4244 021774 012737 000044 001666 MOV #44,$TMP20
4245 022002 013737 177740 001634 MOV @WLOADRS,$TMP3
4246 022010 013737 177742 001636 MOV @WHIADRS,$TMP4
4247 022016 013737 177744 001640 MOV @MMEMERR,$TMP5
4248 022024 010037 001642 MOV R0,$TMP6
4249 022030 005037 001644 CLR $TMP7
4250 022034 010037 001656 MOV R0,$TMP14
4251 022040 062737 000002 001656 ADD #2,$TMP14
4252 022046 005037 001660 CLR $TMP15
4253 022052 011037 001646 MOV (R0),$TMP10
4254 022056 016037 000002 001650 MOV 2(R0),$TMP11
4255 022064 010137 001652 MOV R1,$TMP12
4256 022070 005037 001654 CLR $TMP13
4257 022074 104044 64$: ERROR 44
4258 022076 012737 177777 177744 MOV #-1,@MMEMERR
4259
4260 022104 010002 MOV R0,R2
4261 022106 020102 CMP R1,R2
4262 022110 001002 BNE 2$
4263 022112 000137 021554 JMP CC4
4264 022116 005202 2$: INC R2
4265 022120 020102 CMP R1,R2
4266 022122 001002 BNE 3$
4267 022124 000137 021616 JMP CC7
4268 022130 005202 3$: INC R2
4269 022132 020102 CMP R1,R2
4270 022134 001002 BNE 4$
4271 022136 000137 021660 JMP CC10
4272 022142 000137 022226 4$: JMP CCDONE
4273
4274
4275 022146 011637 001652 CCERR3: MOV (SP),$TMP12 ;REPORT FAILURE TO WRITE
4276 ;THE CORRECT BYTE
4277 022152 010037 001632 MOV R0,$TMP2
4278 022156 005037 001634 CLR $TMP3
4279 022162 010037 001636 MOV R0,$TMP4
4280 022166 062737 000002 001636 ADD #2,$TMP4
4281 022174 005037 001640 CLR $TMP5
4282 022200 011037 001642 MOV (R0),$TMP6
4283 022204 016037 000002 001644 MOV 2(R0),$TMP7
4284 022212 010137 001646 MOV R1,$TMP10
4285 022216 005037 001650 CLR $TMP11
4286 022222 104046 ERROR 46
4287 022224 000207 RTS PC
4288
4289
4290 022226 104416 CCDONE: RSET ;DONE.

```

4291
4292
4293
4294
4295
4296
4297
4298
4299
4300
4301
4302
4303
4304
4305
4306
4307
4308
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318
4319
4320
4321
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331
4332
4333
4334
4335
4336
4337
4338
4339
4340
4341
4342
4343
4344
4345
4346

022230 000004
022232 012737 000010 001676
000021
022240 012737 023026 054230
022246 113737 001502 001626
022254 012737 022532 000114
022262 012737 000014 177746
022270 012700 022526
022274 042700 000003
022300 010001
022302 012710 177777
022306 012760 177777 000002
022314 000240
105011
022320 022710 177400
022324 001403
022326 004737 022744
022332 000404
022334 022760 177777 000002
022342 001371
022344 005201
022346 012710 177777
022352 012760 177777 000002
022360 000240
105011
022364 022710 000377
022370 001403
022372 004737 022744
022376 000404
022400 022760 177777 000002
022406 001371
022410 005201
022412 012710 177777
022416 012760 177777 000002
022424 000240
105011
022430 022760 177400 000002
022436 001403
022440 004737 022744

```

*****
*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.
*
*****
TST21:  SCOPE
MOV     #10,$TIMES      ;;DO 10 ITERATIONS
FF=$^N-1
MOV     #TST22,SKAD     ;SET THE SKAD REGISTER
                        ;IN CASE THE TEST ABORTS.
MOVB   $TSTNM,$TMP0
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
        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
        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
        CLRB   (R1)
        CMP    #177400,2(R0)
        BEQ   FF9
FF8:    JSR    PC,FFERR3
    
```

;FOR SCOPING WITH AN OSCILLOSCOPE.

G 9

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 85
 CEKBD.D.P11 16-MAY-79 08:58 T21 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATAB ZEROES TEST SEQ 0110

4347	022444	000403				BR	FF10		
4348	022446	022710	177777		FF9:	CMP	#-1,(R0)		
4349	022452	001372				BNE	FF8		
4350									
4351	022454	005201			FF10:	INC	R1		;TEST MASK 3.
4352	022456	012710	177777			MOV	#-1,(R0)		
4353	022462	012760	177777	000002		MOV	#-1,2(R0)		
4354	022470	000240				NOP			;FOR SCOPING WITH AN OSCILLOSCOPE.
4355	022472	105011				CLRB	(R1)		
4356	022474	022760	000377	000002		CMP	#377,2(R0)		
4357	022502	001403				BEQ	FF12		
4358	022504	004737	022744		FF11:	JSR	PC,FFERR3		
4359	022510	000403				BR	FF13		
4360	022512	022710	177777		FF12:	CMP	#-1,(R0)		
4361	022516	001372				BNF	FF11		
4362									
4363	022520	000137	023024		FF13:	JMP	FFDONE		
4364									
4365	022524	000000			FFTMP1:	.WORD	0		;TEST AREA.
4366	022526	000000			FFTMP2:	.WORD	0		
4367	022530	000000				.WORD	0		
4368									
4369									
4370	022532	032737	000002	177744	FFERR1:	BIT	#2,@MEMERR		;SHOULD BE A MAIN MEMORY
4371	022540	001002				BNE	1\$;ADDRESS AND CONTROL LINE
4372	022542	000137	054076			JMP	SPUR		;PARITY ERROR.
4373	022546	020137	177740		1\$:	CMP	R1,@LOADRS		;ERROR ADDRESS SHOULD BE
4374	022552	001402				BEQ	FFERR2		;TEST ADDRESS.
4375	022554	000137	054076			JMP	SPUR		
4376	022560	012637	001642		FFERR2:	MOV	(SP)+,\$TMP6		
4377	022564	005037	001664			CLR	\$TMP17		
4378	022570	005726				TST	(SP)+		;RESET THE STACK
4379	022572	012737	000050	001666		MOV	#50,\$TMP20		
4380	022600	013737	177740	001634		MOV	@LOADRS,\$TMP3		
4381	022606	013737	177742	001636		MOV	@HIADRS,\$TMP4		
4382	022614	013737	177744	001640		MOV	@MEMERR,\$TMP5		
4383	022622	010037	001642			MOV	R0,\$TMP6		
4384	022626	005037	001644			CLR	\$TMP7		
4385	022632	010037	001656			MOV	R0,\$TMP14		
4386	022636	062737	000002	001656		ADD	#2,\$TMP14		
4387	022644	005037	001660			CLR	\$TMP15		
4388	022650	011037	001646			MOV	(R0),\$TMP10		
4389	022654	016037	000002	001650		MOV	2(R0),\$TMP11		
4390	022662	010137	001652			MOV	R1,\$TMP12		
4391	022666	005037	001654			CLR	\$TMP13		
4392	022672	104050			64\$:	ERROR	50		
4393	022674	012737	177777	177744		MOV	#-1,@MEMERR		
4394									
4395	022702	010002				MOV	R0,R2		
4396	022704	020102				CMP	R1,R2		
4397	022706	001002				BNE	2\$		
4398	022710	000137	022344			JMP	FF4		
4399	022714	005202			2\$:	INC	R2		
4400	022716	020102				CMP	R1,R2		
4401	022720	001002				BNE	3\$		
4402	022722	000137	022410			JMP	FF7		

```

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) H 9 16-MAY-79 09:11 PAGE 86
CEKBD.P11 16-MAY-79 08:58 T21 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, CPU DATOB ZEROES TEST SEQ 0111

4403 022726 005202 3$: INC R2
4404 022730 020102 CMP R1,R2
4405 022732 001002 BNE 4$
4406 022734 000137 022454 JMP FF10
4407 022740 000137 023024 4$: JMP FFDONE ;HALT???'
4408
4409
4410 022744 011637 001652 FFERR3: MOV (SP),$TMP12 ;REPORT FAILURE TO WRITE
4411 ;THE CORRECT BYTE
4412 022750 010037 001632 MOV R0,$TMP2
4413 022754 005037 001634 CLR $TMP3
4414 022760 010037 001636 MOV R0,$TMP4
4415 022764 062737 000002 001636 ADD #2,$TMP4
4416 022772 005037 001640 CLR $TMP5
4417 022776 011037 001642 MOV (R0),$TMP6
4418 023002 016037 000002 001644 MOV 2(R0),$TMP7
4419 023010 010137 001646 MOV R1,$TMP10
4420 023014 005037 001650 CLR $TMP11
4421 023020 104052 ERROR 52
4422 023022 000207 RTS PC
4423
4424
4425 023024 104416 FFDONE: RSET ;DONE.
4426
4427
4428 ;*****
4429 ;*TEST 22 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATOB ONES TEST*
4430 ;*
4431 ;*THIS IS A TEST OF THE BYTE MASK GENERATION LOGIC. THIS
4432 ;*IS A FOUR BIT MASK USED BY MAIN MEMORY WHEN PERFORMING
4433 ;*A WRITE. IT DESIGNATES WHICH BYTES OF THE TWO WORDS OF
4434 ;*DATA ON THE MAIN MEMORY DATA BUS LINES ARE TO
4435 ;*BE WRITTEN. THIS WILL BE A TEST DOING UNIBUS DATOB REFERENCES TO
4436 ;*THE CACHE. THE DATOB WILL WRITE 377 INTO A BACK ROUND PATTERN
4437 ;*OF ZEROES.
4438 ;*
4439 ;*****
4440 023026 000004 TST22: SCOPE
4441 023030 012737 000010 001676 MOV #10,$TIMES ;:DO 10 ITERATIONS
4442 000022 EE=$TN-1
4443 023036 012737 023714 054230 MOV #TST23,SKAD ;SET THE SKAD REGISTER
4444 ;:IN CASE THE TEST ABORTS.
4445 023044 113737 001502 001626 MOV $TSTNM,$TMP0
4446 023052 104422 MMSKIP
4447 023054 012737 023420 000114 MOV #EEERR1,@#CACHVEC
4448
4449 023062 012700 172340 MOV #KIPAR0,R0 ;SET UP MEMORY MANAGEMENT
4450 ;:TO RELOCATE EVERYTHING
4451 023066 012702 172300 MOV #KIPDR0,R2 ;:THROUGH THE UNIBUS
4452 023072 012703 000007 MOV #7,R3 ;:MAP PASSIVELY TO MEMORY,
4453 023076 005004 CLR R4 ;:BY PASSIVELY IS MEANT
4454 023100 012705 170200 MOV #MAPL00,R5 ;:THAT ADDRESS ARE
4455 ;:RELOCATED TO THEMSELVES.
4456 023104 012722 077406 64$: MOV #77406,(R2)+
4457 023110 010401 MOV R4,R1
4458 023112 072127 000006 ASH #6,R1

```



```

4459 023116 010125      MOV      R1,(R5)+
4460 023120 005025      CLR      (R5)+
4461 023122 010410      MOV      R4,(R0)
4462 023124 062720 170000  ADD      #170000,(R0)+
4463 023130 062704 000200  ADD      #200,R4
4464 023134 077315      SOB      R3,64$
4465 023136 012710 177600  MOV      #177600,(R0)
4466 023142 012712 077406  MOV      #77406,(R2)
4467
4468 023146 012737 000060 172516  MOV      #60,@MMR3      ;TURN ON MEMORY MANAGEMENT
4469 023154 012737 000001 177572  MOV      #1,@MMR0      ;AND THE MAPPING BOX RELOCATION.
4470
4471 023162 012737 000014 177746  MOV      #MOM1,@CONTRL ;FORCE MISSES TO BOTH GROUPS.
4472
4473 023170 012700 023414  MOV      #EETMP2,R0      ;LOCATE THE TEST SPACE.
4474 023174 042700 000003  BIC      #3,R0
4475 023200 010001  MOV      R0,R1
4476
4477 023202 005010      EE1:    CLR      (R0)          ;TEST MASK 0
4478 023204 005060 000002  CLR      2(R0)
4479 023210 000240      NOP
4480 023212 112711 000377  MOVB     #377,(R1)      ;FOR SCOPING WITH AN OSCILLOSCOPE.
4481 023216 022710 000377  CMP      #377,(R0)
4482 023222 001403      BEQ      EE3
4483 023224 004737 023632  EE2:    JSR      PC,EEERR3
4484 023230 000403      BR       EE4
4485 023232 005760 000002  EE3:    TST      2(R0)
4486 023236 001372      BNE      EE2
4487
4488 023240 062701 000001  EE4:    ADD      #1,R1
4489 023244 005010      CLR      (R0)
4490 023246 005060 000002  CLR      2(R0)
4491 023252 000240      NOP
4492 023254 112711 000377  MOVB     #377,(R1)      ;FOR SCOPING WITH AN OSCILLOSCOPE.
4493 023260 022710 177400  CMP      #177400,(R0)
4494 023264 001403      BEQ      EE6
4495 023266 004737 023632  EE5:    JSR      PC,EEERR3
4496 023272 000403      BR       EE7
4497 023274 005760 000002  EE6:    TST      2(R0)
4498 023300 001372      BNE      EE5
4499
4500 023302 062701 000001  EE7:    ADD      #1,R1
4501 023306 005010      CLR      (R0)
4502 023310 005060 000002  CLR      2(R0)
4503 023314 000240      NOP
4504 023316 112711 000377  MOVB     #377,(R1)      ;FOR SCOPING WITH AN OSCILLOSCOPE.
4505 023322 022760 000377 000002  CMP      #377,2(R0)
4506 023330 001403      BEQ      EE9
4507 023332 004737 023632  EE8:    JSR      PC,EEERR3
4508 023336 000402      BR       EE10
4509 023340 005710      EE9:    TST      (R0)
4510 023342 001373      BNE      EE8
4511
4512 023344 062701 000001  EE10:   ADD      #1,R1
4513 023350 005010      CLR      (R0)
4514 023352 005060 000002  CLR      2(R0)
    
```

J 9

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 88
 CEKBD.D.P11 16-MAY-79 08:58 T22 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATAB ONES TEST SEQ 0113

```

4515 023356 000240      NOP
4516 023360 112711 000377  MOVB #377,(R1) ;FOR SCOPING WITH AN OSCILLOSCOPE.
4517 023364 022760 177400 000002  CMP #177400,2(R0)
4518 023372 001403      BEQ EE12
4519 023374 004737 023632  EE11: JSR PC,EEERR3
4520 023400 000402      BR EE13
4521 023402 005710  EE12: TST (R0)
4522 023404 001373      BNE EE11
4523
4524 023406 000137 023712  EE13: JMP EEDONE
4525
4526 023412 000000  EETMP1: .WORD 0
4527 023414 000000  EETMP2: .WORD 0
4528 023416 000000      .WORD 0
4529
4530
4531 023420 032737 000002 177744  EEERR1: BIT #2,@MEMERR ;SHOULD BE A MAIN MEMORY
4532 023426 001002      BNE 1$ ;ADDRESS AND CONTROL LINE
4533 023430 000137 054076      JMP SPUR ;PARITY ERROR.
4534 023434 020137 177740  1$: CMP R1,@LOADRS ;ERROR ADDRESS SHOULD BE
4535 023440 001402      BEQ EEERR2 ;TEST ADDRESS.
4536 023442 000137 054076      JMP SPUR
4537 023446 012637 001642  EEERR2: MOV (SP)+,$TMP6
4538 023452 005037 001664      CLR $TMP17
4539 023456 005726      TST (SP)+ ;RESET THE STACK
4540 023460 012737 000045 001666      MOV #45,$TMP20
4541 023466 013737 177740 001634      MOV @LOADRS,$TMP3
4542 023474 013737 177742 001636      MOV @HIADRS,$TMP4
4543 023502 013737 177744 001640      MOV @MEMERR,$TMP5
4544 023510 010037 001642      MOV R0,$TMP6
4545 023514 005037 001644      CLR $TMP7
4546 023520 010037 001656      MOV R0,$TMP14
4547 023524 062737 000002 001656      ADD #2,$TMP14
4548 023532 005037 001660      CLR $TMP15
4549 023536 011037 001646      MOV (R0),$TMP10
4550 023542 016037 000002 001650      MOV 2(R0),$TMP11
4551 023550 010137 001652      MOV R1,$TMP12
4552 023554 005037 001654      CLR $TMP13
4553 023560 104045 64$: ERROR 45
4554 023562 012737 177777 177744      MOV #-1,@MEMERR
4555
4556 023570 010002      MOV R0,R2
4557 023572 020102      CMP R1,R2
4558 023574 001002      BNE 2$
4559 023576 000137 023240      JMP EE4
4560 023602 005202 2$: INC R2
4561 023604 020102      CMP R1,R2
4562 023606 001002      BNE 3$
4563 023610 000137 023302      JMP EE7
4564 023614 005202 3$: INC R2
4565 023616 020102      CMP R1,R2
4566 023620 001002      BNE 4$
4567 023622 000137 023344      JMP EE10
4568 023626 000137 023712 4$: JMP EEDONE
4569
4570

```

```

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 89
CEKBD.P1 16-MAY-79 08:58 T22 CACHE ADDRESS MEMORY BYTE MASK GENERATOR, UNIBUS DATOB ONES TEST SEQ 0114

```

4571	023632	011637	001652	EEERR3:	MOV	(SP), \$TMP12	:REPORT FAILURE TO WRITE
4572							:THE CORRECT BYTE
4573	023636	010037	001632		MOV	R0, \$TMP2	
4574	023642	005037	001634		CLR	\$TMP3	
4575	023646	010037	001636		MOV	R0, \$TMP4	
4576	023652	062737	000002	001636	ADD	#2, \$TMP4	
4577	023660	005037	001640		CLR	\$TMP5	
4578	023664	011037	001642		MOV	(R0), \$TMP6	
4579	023670	016037	000002	001644	MOV	2(R0), \$TMP7	
4580	023676	010137	001646		MOV	R1, \$TMP10	
4581	023702	005037	001650		CLR	\$TMP11	
4582	023706	104047			ERROR	47	
4583	023710	000207			RTS	PC	
4584							
4585							
4586	023712	104416		EEDONE:	RSET		:DONE.
4587							
4588							
4589							
4590							
4591							
4592							
4593							
4594							
4595							
4596	023714	000004		TST23:	SCOPE		
4597	023716	012737	000010	001676	MOV	#10, \$TIMES	::DO 10 ITERATIONS
4598		000023			HH=\$TN-1		
4599							:SET THE SKAD REGISTER
4600	023724	012737	024614	054230	MOV	#TST24, SKAD	:IN CASE THE TEST ABORTS.
4601							
4602	023732	113737	001502	001626	MOVB	\$TSTNM, \$TMP0	
4603							
4604	023740	104422			MMSKIP		
4605							
4606	023742	012737	024320	000114	MOV	#HHERR1, @#CACHVEC	
4607							
4608							
4609	023750	012700	172340		MOV	#KIPAR0, R0	:SET UP MEMORY MANAGEMENT
4610							:TO RELOCATE EVERYTHING
4611	023754	012702	172300		MOV	#KIPDR0, R2	:THROUGH THE UNIBUS
4612	023760	012703	000007		MOV	#7, R3	:MAP PASSIVELY TO MEMORY,
4613	023764	005004			CLR	R4	:BY PASSIVELY IS MEANT
4614	023766	012705	170200		MOV	#MAPL00, R5	:THAT ADDRESS ARE
4615							:RELOCATED TO THEMSELVES.
4616	023772	012722	077406	64\$:	MOV	#77406, (R2)+	
4617	023776	010401			MOV	R4, R1	
4618	024000	072127	000006		ASH	#6, R1	
4619	024004	010125			MOV	R1, (R5)+	
4620	024006	005025			CLR	(R5)+	
4621	024010	010410			MOV	R4, (R0)	
4622	024012	062720	170000		ADD	#170000, (R0)+	
4623	024016	062704	000200		ADD	#200, R4	
4624	024022	077315			SOB	R3, 64\$	
4625	024024	012710	177600		MOV	#177600, (R0)	
4626	024030	012712	077406		MOV	#77406, (R2)	

```

4627
4628 024034 012737 000060 172516      MOV    #60,@MMR3      ;TURN ON MEMORY MANAGEMENT
4629 024042 012737 000001 177572      MOV    #1,@MMR0      ;AND MAPPING BOX RELOCATION.
4630
4631 024050 012737 000014 177746      MOV    #MOM1,@CONTRL ;FORCE MISSES.
4632
4633 024056 012700 024314          MOV    #HHTMP2,R0    ;LOCATE THE TEST SPACE.
4634 024062 042700 000003          BIC    #3,R0
4635 024066 010001          MOV    R0,R1
4636
4637 024070 012710 177777          HH1:  MOV    #-1,(R0)
4638 024074 012760 177777 000002          MOV    #-1,2(R0)
4639 024102 000240          NOP                                ;FOR SCOPING WITH AN OSCILLOSCOPE.
4640 024104 105011          CLR   (R1)
4641 024106 022710 177400          CMP    #177400,(R0)
4642 024112 001403          BEQ   HH3
4643 024114 004737 024532          HH2:  JSR   PC,HHERR3
4644 024120 000404          BR    HH4
4645 024122 022760 177777 000002          HH3:  CMP    #-1,2(R0)
4646 024130 001371          BNE   HH2
4647
4648 024132 005201          HH4:  INC    R1
4649 024134 012710 177777          MOV    #-1,(R0)
4650 024140 012760 177777 000002          MOV    #-1,2(R0)
4651 024146 000240          NOP                                ;FOR SCOPING WITH AN OSCILLOSCOPE.
4652 024150 105011          CLR   (R1)
4653 024152 022710 000377          CMP    #377,(R0)
4654 024156 001403          BEQ   HH6
4655 024160 004737 024532          HH5:  JSR   PC,HHERR3
4656 024164 000404          BR    HH7
4657 024166 022760 177777 000002          HH6:  CMP    #-1,2(R0)
4658 024174 001371          BNE   HH5
4659
4660 024176 005201          HH7:  INC    R1
4661 024200 012710 177777          MOV    #-1,(R0)
4662 024204 012760 177777 000002          MOV    #-1,2(R0)
4663 024212 000240          NOP                                ;FOR SCOPING WITH AN OSCILLOSCOPE.
4664 024214 105011          CLR   (R1)
4665 024216 122760 177400 000002          CMP   #177400,2(R0)
4666 024224 001403          BEQ   HH9
4667 024226 004737 024532          HH8:  JSR   PC,HHERR3
4668 024232 000403          BR    HH10
4669 024234 022710 177777          HH9:  CMP    #-1,(R0)
4670 024240 001372          BNE   HH8
4671
4672 024242 005201          HH10: INC    R1
4673 024244 012710 177777          MOV    #-1,(R0)
4674 024250 012760 177777 000002          MOV    #-1,2(R0)
4675 024256 000240          NOP                                ;FOR SCOPING WITH AN OSCILLOSCOPE.
4676 024260 105011          CLR   (R1)
4677 024262 022760 000377 000002          CMP   #377,2(R0)
4678 024270 001403          BEQ   HH12
4679 024272 004737 024532          HH11: JSR   PC,HHERR3
4680 024276 000403          BR    HH13
4681 024300 022710 177777          HH12: CMP    #-1,(R0)
4682 024304 001372          BNE   HH11
    
```

```

4683
4684 024306 000137 024612      HH13:  JMP      HHDONE
4685
4686 024312 000000
4687 024314 000000      HHTMP1: .WORD   0
4688 024316 000000      HHTMP2: .WORD   0          ;THE TEST AREA
4689
4690
4691 024320 032737 000002 177744  HHERR1: BIT      #2,@MEMERR      ;SHOULD BE A MAIN MEMORY
4692 024326 001002          BNE      1$          ;ADDRESS AND CONTROL LINE
4693 024330 000137 054076          JMP      SPUR        ;PARITY ERROR.
4694 024334 020137 177740      1$:      CMP      R1,@LOADRS      ;ERROR ADDRESS SHOULD BE
4695 024340 001402          BEQ      HHERR2      ;TEST ADDRESS.
4696 024342 000137 054076          JMP      SPUR
4697 024346 012637 001642      HHERR2: MOV      (SP)+,$TMP6
4698 024352 005037 001664          CLR      $TMP7
4699 024356 005726          TST      (SP)+          ;RESET THE STACK
4700 024360 012737 000051 001666      MOV      #51,$TMP20
4701 024366 013737 177740 001634      MOV      @LOADRS,$TMP3
4702 024374 013737 177742 001636      MOV      @HIADRS,$TMP4
4703 024402 013737 177744 001640      MOV      @MEMERR,$TMP5
4704 024410 010037 001642          MOV      R0,$TMP6
4705 024414 005037 001644          CLR      $TMP7
4706 024420 010037 001656          MOV      R0,$TMP14
4707 024424 062737 000002 001656      ADD      #2,$TMP14
4708 024432 005037 001660          CLR      $TMP15
4709 024436 011037 001646          MOV      (R0),$TMP10
4710 024442 016037 000002 001650      MOV      2(R0),$TMP11
4711 024450 010137 001652          MOV      R1,$TMP12
4712 024454 005037 001654          CLR      $TMP13
4713 024460 104051          64$:     ERROR    51
4714 024462 012737 177777 177744      MOV      #-1,@MEMERR
4715
4716 024470 010002          MOV      R0,R2
4717 024472 020102          CMP      R1,R2
4718 024474 001002          BNE      2$
4719 024476 000137 024132          JMP      HH4
4720 024502 005202          2$:      INC      R2
4721 024504 020102          CMP      R1,R2
4722 024506 001002          BNE      3$
4723 024510 000137 024176          JMP      HH7
4724 024514 005202          3$:      INC      R2
4725 024516 020102          CMP      R1,R2
4726 024520 001002          BNE      4$
4727 024522 000137 024242          JMP      HH10
4728 024526 000137 024612          4$:      JMP      HHDONE
4729
4730
4731 024532 011637 001652      HHERR3: MOV      (SP),$TMP12      ;REPORT FAILURE TO WRITE
4732                                ;THE CORRECT BYTE
4733 024536 010037 001632          MOV      R0,$TMP2
4734 024542 005037 001634          CLR      $TMP3
4735 024546 010037 001636          MOV      R0,$TMP4
4736 024552 062737 000002 001636      ADD      #2,$TMP4
4737 024560 005037 001640          CLR      $TMP5
4738 024564 011037 001642          MOV      (R0),$TMP6
    
```

4739 024570 016037 000002 001644
4740 024576 010137 001646
4741 024602 005037 001650
4742 024606 104053
4743 024610 000207
4744
4745

MOV 2(R0), \$TMP7
MOV R1, \$TMP10
CLR \$TMP11
ERROR 53
RTS PC

4746 024612 104416

HHDONE: RSET ; DONE.

4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771

*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 INVLDATER LOGIC
*WORKED NO PARITY ERRORS CAN OCCUR. BUT IF THIS INVALIDATER
*FAILED THERE IS AN EXTREMELY HIGH PROBABILITY FOR THE
*OCCURENCE OF A CACHE DATA OR CACHE ADDRESS PARITY ERROR.
*IN FACT IF THE INVALIDATER 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.
*

4772 024614 000004
4773 000024

TST24: SCOPE
DD \$TN-1

4774
4775 024616 012737 025050 054230
4776
4777 024624 113737 001502 001626
4778 024632 012737 054076 000114
4779

MOV #TST25, SKAD ;SET THE SKAD REGISTER
;IN CASE THE TEST ABORTS.
MOV \$TSTNM, \$TMP0
MOV #SPUR, @#CACHVEC ;INITIALLY EXPECT NO ERRORS.

4780 024640 032737 010000 177570
4781 024646 001002
4782 024650 000177 027354
4783

BIT #SW12, @#SWR ;SEE IF THE USER HAS CHOSEN
BNE DD1 ;TO RUN THIS TEST, SW12=1.
JMP @SKAD ;NO, SO GO TO NEXT TEST.

4784 024654 012737 025006 000114 DD1:
4785
4786 024662 013737 000024 025036
4787 024670 012737 024710 000024
4788 024676 005037 025040
4789

MOV #DDPER, @#CACHVEC ;YES, SO SET UP THE PARITY
;ERROR VECTOR.
MOV @#24, DDTMP ;SAVE THE OLD CONTENTS
MOV #DDPD, @#24 ;OF THE PWER FAIL TRAP
CLR DDCNTR ;VECTOR AND RESET THIS
;VECTOR. CLEAR AN ERROR COUNT.

4790 024702 104400
4791 024704 065536
4792 024706 000777
4793

TYPE PDMSG1 ;TELL THE USER TO POWER
.WORD ;DOWN.
BR . ;WAIT, SHOULD THIS
;WAIT TIME OUT????

4794 024710 000240

DDPD: NOP ;FOR SCOPE SYNC.

```

4795 024712 012737 024722 000024      MOV      #DDPV, @#24      ;POWER DOWN ROUTINE
4796 024720 000777                    BR          ;JUST SET UP FOR POWER UP.
4797 024722 012706 001500      DDPV:  MOV      #STACK, SP  ;RESET THE STACK POINTER
4798 024726 013737 025036 000024      MOV      DDTMP, @#24     ;RESET POWER FAIL VECTOR.
4799 024734 005000                    CLR      R0              ;SET UP FOR SCAN.
4800 024736 012701 001000                    MOV      #1000, R1
4801 024742 005720      1$:  TST      (R0)+
4802 024744 077102                    SOB      R1, 1$
4803 024746 013737 025036 000024      DDPU1: MOV      DDTMP, @#24     ;RESET THE POWER FAIL VECTOR.
4804 024754 005737 025040                    TST      DDCNTR          ;WERE THERE ANY ERRORS?
4805 024760 001004                    BNE      DDPU2
4806 024762 104400                    TYPE
4807 024764 065714      .WORD  PDMSG2
4808 024766 000137 025042      JMP      DDDONE          ;NO
4809
4810 024772                    DDPU2:                    ;REPORT ERROR SUMMARY
4811 024772 013737 025040 001632      MOV      DDCNTR, $TMP2
4812 025000 104054      1$:  ERROR  54
4813 025002 000137 025042      JMP      DDDONE
4814
4815 025006 032737 000360 177744      DDPER: BIT      #360, @MEMERR ;THE ERROR SHOULD BE
4816 025014 001406                    BEQ      DDPER1          ;A CACHE ADDRESS OR CACHE
4817 025016 012737 177777 177744      MOV      #-1, MEMERR    ;DATA PARITY ERROR
4818 025024 005237 025040                    INC      DDCNTR
4819 025030 000002                    RTI
4820
4821 025032 000137 054076      DDPER1: JMP      SPUR
4822
4823 025036 000000      DDTMP. .WORD  0          ;STORAGE FOR POWER FAIL
4824 025040 000000                    DDCNTR: .WORD  0          ;VECTORS OLD PC
4825 025042 104416                    DDDONE: RSET
4826 025044 012706 001500      MOV      #STACK, SP
4827
4828
4829
4830      ;:*****
4831      ;*TEST 25      CACHE DATA MULTIPLEXER, CDMX, TEST
4832      ;*
4833      ;*THIS TEST PUTS DIFFERENT PATTERNS OF DATA AT THE INPUTS
4834      ;*OF THE CDMX AND TESTS FOR PROPER SELECTION AND GOOD DATA.
4835      ;*
4836      ;:*****
4837 025050 000004      TST25: SCOPE
4838 025052 012737 000010 001676      MOV      #10, $TIMES    ;;DO 10 ITERATIONS
4839
4840 025060 012737 026156 054230      MOV      #TST26, SKAD   ;SET THE SKAD REGISTER
4841
4842 025066 012737 054076 000114      MOV      #SPUR, @#CACHVEC ;PREPARE FOR UNEXPECTED ERRORS.
4843 025074 113737 001502 001626      MOV      $STNM, $TMP0
4844 025102 012705 000006                    MOV      #6, R5          ;INITIALIZE
4845 025106 012737 000004 026130      MOV      #4, JJCNT
4846 025114 012700 026146      MOV      #JJTMP2, R0
4847 025120 042700 176002      BIC      #176002, R0
4848 025124 012701 140000      MOV      #TESTR1, R1
4849 025130 060001      ADD      R0, R1
4850 025132 012702 142000      MOV      #TESTR2, R2

```

4851	025136	060002				ADD	R0,R2	
4852	025140	012703	144000			MOV	#TESTR3,R3	
4853	025144	060003				ADD	R0,R3	
4854	025146	012704	026134			MOV	#JJPAT2,R4	
4855								
4856	025152	012737	125252	026132		MOV	#125252,JJPAT1	:JJPAT1 CONTAINS THE DATA
4857								:WHICH WILL ENTER THE
4858								:MAIN MEMORY EVEN INPUTS
4859								:TO THE CDMX. INITIALLY
4860								:THIS WILL BE 125252
4861	025160	012737	052525	026134		MOV	#52525,JJPAT2	:DATA FOR MAIN MEMORY ODD
4862								:WORD INPUT TO CDMX
4863	025166	005037	026136			CLR	JJPAT3	:GROUP 0 DATA INPUTS TO CDMX.
4864	025172	012737	177777	026140		MOV	#-1,JJPAT4	:GROUP 1 DATA INPUTS TO CDMX.
4865	025200	012737	025200	001512	JJ1:	MOV	#JJ1,\$LPERR	
4866	025206	013713	026132			MOV	JJPAT1,(R3)	:WRITE THE MAIN MEMORY
4867	025212	013763	026134	000002		MOV	JJPAT2,2(R3)	:EVEN AND ODD WORD PATTERNS
4868								
4869	025220	012737	000034	177746		MOV	#SOMOM1,@#CONTRL	:WRITE THE GROUP ZERO
4870	025226	013711	026136			MOV	JJPAT3,(R1)	:PATTERN
4871	025232	013761	026136	177776		MOV	JJPAT3,-2(R1)	
4872	025240	013761	026136	000002		MOV	JJPAT3,2(R1)	
4873	025246	005711				TST	(R1)	
4874	025250	012737	000054	177746		MOV	#S1MOM1,@#CONTRL	:WRITE THE GROUP ONE PATTERN
4875	025256	013712	026140			MOV	JJPAT4,(R2)	
4876	025262	013762	026140	177776		MOV	JJPAT4,-2(R2)	
4877	025270	013762	026140	000002		MOV	JJPAT4,2(R2)	
4878	025276	005712				TST	(R2)	
4879								
4880	025300	005037	177746			CLR	@#CONTRL	
4881	025304	000240				NOP		
4882	025306				JJ2:			
4883	025306	000240				NOP		
4884	025310	016100	000000			MOV	0(R1),R0	
4885	025314	032737	000010	177752		BIT	#10,@#HITMIS	:MUST BE A HIT!
4886	025322	001011				BNF	JJ3	
4887	025324	012737	000000	001630		MOV	#0,\$TMP1	
4888	025332	010137	001632			MOV	R1,\$TMP2	
4889	025336	062737	000000	001632		ADD	#0,\$TMP2	
4890	025344	104001			66\$:	ERROR	1	
4891	025346	020037	026136		JJ3:	CMP	R0,JJPAT3	
4892	025352	001406				BEQ	65\$	
4893	025354	012737	025366	001630		MOV	#64\$,\$TMP1	
4894	025362	010037	001632			MOV	R0,\$TMP2	
4895	025366	104005			64\$:	ERROR	5	
4896	025370				65\$:			
4897	025370	012737	025376	001512		MOV	#JJ4,\$LPERR	
4898	025376				JJ4:			
4899	025376	000240				NOP		
4900	025400	016100	000002			MOV	2(R1),R0	
4901	025404	032737	000010	177752		BIT	#10,@#HITMIS	:MUST BE A HIT.
4902	025412	001011				BNE	JJ5	
4903	025414	012737	000000	001630		MOV	#0,\$TMP1	
4904	025422	010137	001632			MOV	R1,\$TMP2	
4905	025426	062737	000002	001632		ADD	#2,\$TMP2	
4906	025434	104001			66\$:	ERROR	1	

4907	025436	020037	026136	JJ5:	CMP	R0,JJPAT3	
4908	025442	001406			BEQ	65\$	
4909	025444	012737	025456	001630	MOV	#64\$,STMP1	
4910	025452	010037	001632		MOV	R0,STMP2	
4911	025456	104005		64\$:	ERROR	5	
4912	025460			65\$:			
4913	025460	012737	025466	001512	MOV	#JJ6,\$LPERR	
4914	025466			JJ6:			
4915	025466	000240			NOP		
4916	025470	016200	000000		MOV	0(R2),R0	
4917	025474	032737	000010	177752	BIT	#10,@#HITMIS	;MUST BE A HIT.
4918	025502	001011			BNE	JJ7	
4919	025504	012737	000001	001630	MOV	#1,STMP1	
4920	025512	010237	001632		MOV	R2,STMP2	
4921	025516	062737	000000	001632	ADD	#0,STMP2	
4922	025524	104001		66\$:	ERROR	1	
4923	025526	020037	026140	JJ7:	CMP	R0,JJPAT4	
4924	025532	001406			BEQ	65\$	
4925	025534	012737	025546	001630	MOV	#64\$,STMP1	
4926	025542	010037	001632		MOV	R0,STMP2	
4927	025546	104006		64\$:	ERROR	6	
4928	025550			65\$:			
4929	025550	012737	025556	001512	MOV	#JJ8,\$LPEPR	
4930	025556			JJ8:			
4931	025556	000240			NOP		
4932	025560	016200	000002		MOV	2(R2),R0	
4933	025564	032737	000010	177752	BIT	#10,@#HITMIS	;MUST BE A HIT
4934	025572	001011			BNE	JJ9	
4935	025574	012737	000001	001630	MOV	#1,STMP1	
4936	025602	010237	001632		MOV	R2,STMP2	
4937	025606	062737	000002	001632	ADD	#2,STMP2	
4938	025614	104001		66\$:	ERROR	1	
4939	025616	020037	026140	JJ9:	CMP	R0,JJPAT4	
4940	025622	001406			BEQ	65\$	
4941	025624	012737	025636	001630	MOV	#64\$,STMP1	
4942	025632	010037	001632		MOV	R0,STMP2	
4943	025636	104006		64\$:	ERROR	6	
4944	025640			65\$:			
4945	025640	012737	025646	001512	MOV	#JJ10,\$LPERR	
4946	025646	000240		JJ10:	NOP		
4947	025650	012737	000014	177746	MOV	#M1M0,@#CONTRL	;CHECK MAIN MEMORY DATA
4948	025656	011300			MOV	(R3),R0	;EVEN WORD
4949	025660	020037	026132		CMP	R0,JJPAT1	
4950	025664	001403			BEQ	1\$	
4951	025666	010037	001632		MOV	R0,STMP2	
4952	025672	104007			ERROR	7	
4953	025674	012737	025702	001512	1\$:	MOV	#JJ11,\$LPERR
4954	025702	016300	000002	JJ11:	MOV	2(R3),R0	;CHECK MAIN MEMORY EVEN
4955	025706	020037	026134		CMP	R0,JJPAT2	;WORD
4956	025712	001403			BEQ	JJ12	
4957	025714	010037	001632		MOV	R0,STMP2	
4958	025720	104010		1\$:	ERROR	10	
4959							
4960	025722	005037	177746	JJ12:	CLR	@#CONTRL	
4961	025726	020427	026140		CMP	R4,#JJPAT4	;NOW GET EVERY PERMUTATION
4962	025732	001011			BNE	JJ13	;OF THE FOUR TEST PATTERNS:

```

4963
4964 025734 011437 026142      MOV      (R4),JJPAT5      ;125252,052525,177777 AND
4965 025740 013714 026134      MOV      JJPAT2,(R4)     ;000000 INTO JJPAT1, JJPAT2,
4966 025744 012704 026134      MOV      #JJPAT2,R4     ;JJPAT3 AND JJPAT4 AND
4967 025750 013714 026142      MOV      JJPAT5,(R4)    ;REPEAT THE TEST.
4968 025754 000406      BR      JJ14
4969
4970 025756 012437 026142      JJ13:   MOV      (R4)+,JJPAT5
4971 025762 011464 177776      MOV      (R4),-2(R4)
4972 025766 013714 026142      MOV      JJPAT5,(R4)
4973
4974 025772 005305      JJ14:   DEC      R5
4975 025774 001402      BEQ     1$
4976 025776 000137 025200      JMP     JJ1
4977 026002 012705 000006      1$:    MOV      #6,R5
4978 026006 013737 026132 026142      MOV      JJPAT1,JJPAT5
4979 026014 005337 026130      DEC     JJCNT
4980
4981 026020 023727 026130 000003      CMP     JJCNT,#3
4982 026026 001010      BNE    JJ15
4983 026030 013737 026134 026132      MOV     JJPAT2,JJPAT1
4984 026036 013737 026142 026134      MOV     JJPAT5,JJPAT2
4985 026044 000137 025200      JMP     JJ1
4986
4987 026050 023727 026130 000002      JJ15:  CMP     JJCNT,#2
4988 026056 001010      BNE    JJ16
4989 026060 013737 026136 026132      MOV     JJPAT3,JJPAT1
4990 026066 013737 026142 026136      MOV     JJPAT5,JJPAT3
4991 026074 000137 025200      JMP     JJ1
4992
4993 026100 023727 026130 000001      JJ16:  CMP     JJCNT,#1
4994 026106 001023      BNE    JJ17
4995 026110 013737 026140 026132      MOV     JJPAT4,JJPAT1      ;DONE?
4996 026116 013737 026142 026140      MOV     JJPAT5,JJPAT4
4997 026124 000137 025200      JMP     JJ1
4998
4999 026130 000000      JJCNT: .WORD 0           ;COUNTER USED TO GENERATE
5000                                     ;PERMUTATIONS.
5001 026132 000000      JJPAT1: .WORD 0         ;MAIN MEMORY EVEN WORD DATA PATTERN
5002 026134 000000      JJPAT2: .WORD 0         ;MAIN MEMORY ODD WORD DATA PATTERN
5003 026136 000000      JJPAT3: .WORD 0         ;GROUP 0 DATA PATTERN
5004 026140 000000      JJPAT4: .WORD 0         ;GROUP 1 DATA PATTERN
5005 026142 000000      JJPAT5: .WORD 0         ;TEMPORARY STORAGE
5006
5007 026144 000000      JJTMP1: .WORD 0         ;TEST AREA, SO CODE WON'T
5008 026146 000000 000000 000000      JJTMP2: .WORD 0,0,0,0   ;OVER LAP THE HITS OF
5009 026154 000000                                     ;THE TEST WORDS.
5010
5011
5012 026156      JJ17:                                     ;DONE!
5013
5014      ;*****
5015      ;*TEST 26      CACHE DATA MEMORY ADDRESS DRIVERS TEST
5016      ;*
5017      ;*THIS TEST PERFORMS A DUAL ADDRESS TEST ON THE
5018      ;*CACHE DATA MEMORIES OF BOTH GROUPS.

```

```

5019
5020
5021 026156 000004
5022 026160 012737 000010 001676
5023
5024 026166 012737 026666 054230
5025
5026 026174 012737 054076 000114
5027 026202 113737 001502 001626
5028
5029 026210 012737 000001 026660 GG1:
5030 026216 012737 000054 026662
5031 026224 012737 000034 026664
5032
5033
5034 026232 012700 026232 GG2:
5035 026236 012701 001000
5036 026242 013737 026662 177746 GG3:
5037 026250 005760 002000
5038 026254 013737 026664 177746
5039 026262 005720
5040 026264 077112
5041 026266 013700 026662
5042 026272 042700 177717
5043 026276 010037 177746
5044 026302 012701 140000
5045 026306 012700 001000
5046 026312 012737 026320 001512
5047 026320 000240 GG4:
5048 026322 005011
5049 026324 005711
5050 026326 005711
5051 026330 032737 000010 177752
5052 026336 001006
5053 026340 013737 026660 001630
5054 026346 010137 001632
5055 026352 104001 1$:
5056 026354 005721 2$:
5057 026356 077020
5058 026360 013700 026664
5059 026364 042700 177717
5060 026370 010037 177746
5061
5062 026374 012701 140000
5063 026400 012700 001000
5064 026404 012737 026412 001512
5065 026412 000240 GG5:
5066 026414 010111
5067 026416 005721
5068 026420 077004
5069
5070 026422 012701 140000
5071 026426 012700 001000
5072 026432 012737 026440 001512
5073 026440 000240 GG6:
5074 026442 011102

```

```

: *
: *****
TST26: SCOPE
MOV #10,$TIMES ;:DO 10 ITERATIONS
;SET THE SKAD REGISTER
MOV #TST27,SKAD ;: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 GGS,@#CONTRL ;:HITS IN THE GROUP NOT
TST 2000(R0) ;:BEING TESTED AND MISSES
MOV GGM,@#CONTRL ;:IN THE GROUP BEING TESTED.
TST (R0)+
SOB R1,GG3
MOV GGS,R0 ;:MAKE THE TEST AREA
BIC #177717,R0 ;:HITS IN THE GROUP
MOV R0,@#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 GGM,R0 ;:FROM HERE ON SELECT
BIC #177717,R0 ;:THE GROUP NOT BEING
MOV R0,@#CONTRL ;:TESTED
MOV #TESTR1,R1
MOV #1000,R0
MOV #GG5,$LPERR
GG5: NOP
;WRITE #ADDRESS INTO @#ADDRESS.
MOV R1,(R1)
TST (R1)+
SOB R0,GG5
MOV #TESTR1,R1
MOV #1000,R0
MOV #GG6,$LPERR
GG6: NOP
MOV (R1),R2 ;:READ BACK THE ADDRESS

```

5075 026444 032737 000010 177752
5076 026452 001006
5077 026454 013737 026660 001630

BIT #10, @HITMIS
BNE GG7
MOV GGFLG1, \$TMP1

```

5078 026462 010137 001632      MOV    R1,$TMP2
5079 026466 104001      1$:   ERROR    1
5080
5081 026470 020102      GG7:  CMP     R1,R2      ;DOES @#ADDRESS CONTAIN
5082 026472 001412      BEQ    GG8          ;#ADDRESS
5083
5084 026474 013737 026660 001630      MOV    GGFLG1,$TMP1
5085 026502 010137 001632      MOV    R1,$TMP2
5086 026506 010237 001634      MOV    R2,$TMP3
5087 026512 010137 001636      MOV    R1,$TMP4
5088 026516 104016      1$:   ERROR    16
5089
5090 026520 005121      GG8:  COM    (R1)+      ;COMPLIMENT DATA
5091 026522 077032      SOB    R0,GG6      ;LOOP FOR NEXT ADDRESS.
5092 026524 012701 140000      MOV    #TESTR1,R1
5093 026530 012700 001000      MOV    #1000,R0
5094 026534 012737 026542 001512      MOV    #GG9,$LPERR
5095 026542 000240      CG9:  NOP
5096 026544 011102      MOV    (R1),R2      ;GO BACK AND CHECK
5097 026546 032737 000010 177752      BIT    #10,@#HITMIS ;COMPLIMENTED DATA
5098 026554 001006      BNE    GG10
5099 026556 013737 026660 001630      MOV    GGFLG1,$TMP1
5100 026564 010137 001632      MOV    R1,$TMP2
5101 026570 104001      1$:   ERROR    1
5102
5103
5104 026572 010103      CG10: MOV    R1,R3      ;IS COMPLIMENT DATA CORRECT?
5105 026574 005103      COM    R3
5106 026576 020302      CMP    R3,R2
5107 026600 001412      BEQ    GG11
5108 026602 013737 026660 001630      MOV    GGFLG1,$TMP1
5109 026610 010337 001632      MOV    R3,$TMP2
5110 026614 010237 001634      MOV    R2,$TMP3
5111 026620 010137 001636      MOV    R1,$TMP4
5112 026624 104016      1$:   ERROR    16
5113
5114 026626 005721      GG11: TST    (R1)+      ;TEST NEXT LOCATION
5115 026630 077034      SOB    R0,GG9
5116
5117 026632 012737 000034 026662      MOV    #SOMOM1,GGGS ;GO BACK AND RUN
5118 026640 012737 000054 026664      MOV    #S1MOM1,GGGM ;TEST IN GROUP 0.
5119 026646 005337 026660      DEC    GGFLG1
5120 026652 001005      BNE    GG12
5121 026654 000137 026232      JMP    GG2
5122
5123 026660 000000      GGFLG1: .WORD 0      ;GROUP BEING TESTED, 0 OR 1.
5124
5125 026662 000000      GGGs:  .WORD 0      ;CACHE CONTROL REGISTER
5126 026664 000000      GGGM:  .WORD 0      ;PATTERNS
5127
5128 026666      GG12:
5129
5130
5131
5132
5133

```

```

*****
*TEST 27      CACHE DATA MEMORY COUNT PATTERN TEST
*
*THIS TEST RUNS A COUNT PATTERN THROUGH EACH LOCATION

```

```

5134 ;*OF THE CACHE DATA MEMORY FOR EACH GROUP.
5135 ;*
5136 ;*****
5137 026666 000004 TST27: SCOPE
5138 026670 012737 000010 001676 MOV #10,$TIMES ;;DO 10 ITERATIONS
5139 MOV #TST30,SKAD ;;SET THE SKAD REGISTER
5140 026676 012737 027646 054230 MOV #TST30,SKAD ;IN CASE THE TEST ABORTS.
5141
5142 026704 012737 054076 000114 MOV #SPUR,@#CACHVEC
5143 026712 113737 001502 001626 MOVB $TSTNM,$TMP0
5144
5145 026720 012737 000001 027354 LL1: MOV #1,LLFLG1 ;TEST GROUP ONE FIRST
5146 026726 012737 000044 027362 MOV #S1M0,LLGS ;S1M0 AND SOM1 ARE PATTERNS
5147 026734 012737 000030 027364 MOV #SOM1,LLGM ;WHICH WILL BE LOADED INTO
5148 026742 012737 026742 001512 LL2: MOV #LL2,$LPERR ;THE CACHE CONTROL REGISTER.
5149 026750 012737 054076 000114 MOV #SPUR,@#CACHVEC
5150 026756 012700 026742 MOV #LL2,R0 ;MAKE THIS CODE, LOCATIONS
5151 026762 012701 001000 MOV #1000,R1 ;LL2 THROUGH LL2+2000 (OCT)
5152 ;HITS IN THE CACHE GROUP
5153 026766 013737 027364 177746 LL3: MOV LLGM,@#CONTRL ;NOT BEING TESTED, AND MISSES
5154 026774 005710 TST (R0) ;TO THE CACHE GROUP BEING
5155 026776 013737 027362 177746 MOV LLGS,@#CONTRL ;TESTED.
5156 027004 005760 002000 TST 2000(R0)
5157 027010 062700 000002 ADD #2,R0
5158 027014 077114 SOB R1,LL3
5159
5160 027016 012701 140000 MOV #TESTR1,R1 ;MAKE THE MEMORY TEST AREA
5161 027022 012700 001000 MOV #1000,R0 ;HITS IN THE GROUP BEING
5162 027026 012737 027050 001512 MOV #1$, $LPERR ;TESTED.
5163 027034 013702 027362 MOV LLGS,R2
5164 027040 042702 177717 BIC #177717,R2
5165 027044 010237 177746 MOV R2,@#CONTRL
5166 027050 005011 1$: CLR (R1)
5167 027052 005711 TST (R1)
5168 027054 005721 TST (R1)+
5169 027056 032737 000010 177752 BIT #10,@#HITMIS
5170 027064 001011 BNE 3$
5171 027066 013737 027354 001630 MOV LLFLG1,$TMP1
5172 027074 011137 001632 MOV (R1),$TMP2
5173 027100 062737 177776 001632 ADD #-2,$TMP2
5174 027106 104001 2$: ERROR 1
5175 027110 077021 3$: SOB R0,1$
5176 027112 013700 027364 MOV LLGM,R0 ;FROM NOW ON SELECT
5177 027116 042700 177717 BIC #177717,R0 ;THE GROUP NOT BEING
5178 027122 010037 177746 MOV R0,@#CONTRL ;TESTED
5179
5180 027126 012701 140000 MOV #TESTR1,R1 ;INITIALIZE FOR TEST.
5181 027132 012700 001000 MOV #1000,R0 ;COUNTER.
5182 027136 005002 LL4: CLR R2 ;DATA PATTERN WRITTEN
5183 027140 005003 CLR R3 ;LOGICAL 'OR' OF BAD DATA
5184 027142 012704 177777 MOV #177777,R4 ;LOGICAL 'AND' OF BAD DATA
5185 027146 005005 CLR R5 ;DATA PATTERN READ
5186 027150 005037 027366 CLR LLCNT1 ;NUMBER OF LOCATIONS WHICH FAIL.
5187 027154 005037 027356 CLR LLFLG2 ;ERROR IN GROUP FLAG
5188 027160 012737 027166 001512 MOV #LL5,$LPERR
5189 027166 005037 027360 LL5: CLR LLFLG4 ;ERROR IN TESTED WORD FLAG.
    
```

```

5190 027172 000240      NOP
5191 027174 010211      MOV      R2,(R1)
5192 027176 011105      MOV      (R1),R5
5193 027200 032737 000010 177752      BIT      #10,@#HITMIS
5194 027206 001006      BNE      LL6
5195 027210 013737 027354 001630      MOV      LLFLG1,$TMP1
5196 027216 010137 001632      MOV      R1,$TMP2
5197 027222 104001      1$:      ERROR 1
5198 027224 020205      LL6:      CMP      R2,R5
5199 027226 001402      BEQ      LL7
5200 027230 000137 027600      JMP      LLERR2
5201
5202 027234      LL7:
5203
5204 027234 005737 027360      TST      LLFLG4
5205 027240 001405      BEQ      LL8
5206 027242 005237 027366      INC      LLCNT1
5207 027246 012737 177777 027356      MOV      #-1,LLFLG2
5208 027254 062701 000002      LL8:      ADD      #2,R1
5209 027260 077036      SOB      R0,LL5
5210
5211 027262 005737 027356      TST      LLFLG2
5212 027266 001417      BEQ      LL9
5213 027270 112737 000013 001516      MOV      #13,$ITEMB
5214 027276 013737 027354 001630      MOV      LLFLG1,$TMP1
5215 027304 010437 001632      MOV      R4,$TMP2
5216 027310 010337 001634      MOV      R3,$TMP3
5217 027314 013737 027366 001636      MOV      LLCNT1,$TMP4
5218 027322 004737 054744      JSR      PC,ERTYPE
5219
5220 027326 012737 000044 027364      LL9:      MOV      #S1M0,LLGM
5221 027334 012737 000030 027362      MOV      #S0M1,LLGS
5222 027342 005337 027354      DEC      LLFLG1
5223 027346 001137      BNE      LL 0
5224 027350 000137 026742      JMP      LL2
5225
5226 027354 000000      LLFLG1: .WORD 0
5227 027356 000000      LLFLG2: .WORD 0
5228
5229 027360 000000      LLFLG4: .WORD 0
5230
5231 027362 000000      LLGS:   .WORD 0
5232 027364 000000      LLGM:   .WORD 0
5233
5234 027366 000000      LLCNT1: .WORD 0
5235
5236 027370 000000      LLMER:  .WORD 0
5237
5238 027372 000000      LLTMP1: .WORD 0
5239
5240 027374 013737 177744 027370      LLERR1: MOV      @MEMERR,LLMER
5241 027402 012737 004100 027372      MOV      #4100,LLTMP1
5242 027410 005737 027354      TST      LLFLG1
5243 027414 001403      BEQ      1$
5244 027416 012737 004200 027372      MOV      #4200,LLTMP1
5245 027424 023737 027372 027370      1$:      CMP      LLTMP1,LLMER

```

;FOR SCOPING WITH AN OSCILLOSCOPE.

;GOOD DATA

;BAD DATA BUT NO TRAP OR
;ABORT OCCURRED!
;DECREMENT THE COUNT PATTERN
;AND LOOP IF NOT DONE
;IF THERE WAS AN ERROR
;IN THE WORD JUST TESTED
;INCREMENT LLCNT1
;AND SET ERROR IN GROUP FLAG.
;GO TO NEXT WORD.

;DONE WITH THAT GROUP,
;SEE IF THERE WERE
;ANY ERRORS. IF SO THEN
;PRINT AN ERROR SUMMARY
;FOR THAT GROUP.

;TEST THE OTHER GROUP, 0.

;DONE?

;GROUP BEING TESTED, 1 OR 0.
;ERROR OCCURRED IN GROUP FLAG.

;ERROR OCCURRED IN WORD FLAG.

;PATTERNS FOR CONTROL REGISTER

;GROUP ERROR COUNT

;TEMPORARY STORAGE FOR
;THE CACHE ERROR REGISTER.

;COME HERE ON PARITY
;ABORT OR TRAP.
;TESTING GROUP 1 OR 0?

;WAS THE ERROR EXPECTED?

```
5246 027432 001402 BEQ 2$
5247 027434 000137 054076 JMP SPUR ;NO!
5248
5249 027440 020137 177740 2$: CMP R1,@#LOADRS ;WAS THAT ADDRESS EXPECTED?
5250 027444 001402 BEQ 3$
5251 027446 000137 054076 JMP SPUR ;NO.
5252
5253 027452 012737 177777 027360 3$: MOV #-1,LLFLG4 ;SET WORD ERROR FLAG
5254 027460 050203 BIS R2,R3 ;DO 'OR' OF FAILING DATA
5255 027462 005102 COM R2
5256 027464 040204 BIC R2,R4 ;DO 'AND' OF FAILING DATA
5257 027466 005102 COM R2
5258 027470 011637 001630 MOV (SP),$TMP1
5259 027474 022626 CMP (SP)+,(SP)+
5260 027476 013737 027354 001632 MOV LLFLG1,$TMP2
5261 027504 010237 001634 MOV R2,$TMP3
5262 027510 010137 001644 MOV R1,$TMP7
5263 027514 013737 177740 001636 MOV @#LOADRS,$TMP4
5264 027522 013737 177742 001640 MOV @#HIADRS,$TMP5
5265 027530 042737 140000 001640 BIC #140000,$TMP5
5266 027536 013737 027370 001642 MOV LLMER,$TMP6
5267 027544 104011 ERROR 11 ;REPORT ERROR.
5268
5269 027546 012737 027560 000114 MOV #LLERR3,@#CACHVEC ;BEFORE CONTINUING THE
5270 ;BAD PARITY IN THE WORD
5271 ;BEING TESTED MUST BE
5272 ;DEALT WITH!
5273 027554 005011 CLR (R1) ;THIS INSTRUCTION CLR (R1)
5274 027556 005711 TST (R1) ;SHOULD TRAP!
5275
5276 027560 012737 177777 177744 LLERR3: MOV #-1,@#MEMERR ;CLR THE ERROR REGISTER
5277 027566 012737 027374 000114 MOV #LLERR1,@#CACHVEC ;RESTORE THE PARITY ERROR
5278 027574 000137 027234 JMP LL7 ;VECTOR AND CONTINUE.
5279
5280 027600 012737 177777 027360 LLERR2: MOV #-1,LLFLG4 ;BAD DATA WAS READ BUT
5281 ;NO TRAP OR ABORT OCCURRED!
5282 027606 050203 BIS R2,R3 ;'OR' BAD DATA
5283 027610 005102 COM R2
5284 027612 040204 BIC R2,R4 ;'AND' BAD DATA
5285 027614 005102 COM R2
5286 027616 013737 027354 001630 MOV LLFLG1,$TMP1
5287 027624 010137 001634 MOV R1,$TMP3
5288 027630 010237 001636 MOV R2,$TMP4
5289 027634 010537 001640 MOV R5,$TMP5
5290
5291 027640 104012 1$: ERROR 12 ;REPORT ERROR.
5292
5293 027642 000137 027234 JMP LL7 ;CONTINUE TEST.
5294 027646 LL10:
5295
5296
5297 ;*****
5298 ;*TEST 30 CACHE DATA MEMORY PARITY CHECKERS LOW BYTE TEST
5299 ;*
5300 ;*THIS IS A TEST OF THE TWO CACHE DATA MEMORY PARITY
5301 ;*CHECKERS FOR THE LOW BYTE, ONE FOR EACH GROUP. THE
```



```

5302      *MAINTENANCE REGISTER IS USED TO FORCE A PARITY A
5303      *PARITY ERROR AT EVERY DATA PATTERN WHICH HAS A ONE
5304      *PARITY BIT. NOTE THAT THE CACHE DATA MEMORY PARITY HAS,
5305      *EFFECTIVELY, ODD PARITY. THE MAINTENANCE FUNCTION ON THE
5306      *CACHE DATA MEMORY PARITY CHECKERS HAS THE EFFECT OF
5307      *FORCING THE PARITY BIT OF THE BYTE BEING CHECKED TO
5308      *ZERO. THIS MEANS THAT ONCE THIS MAINTENANCE FUNCTION
5309      *IS ENABLED THE ERROR WILL OCCUR ON A SUBSEQUENT
5310      *READ OF A BYTE WITH A ONE PARITY BIT, THAT IS
5311      *BYTES WITH ZERO PARITY BITS WILL NOT CAUSE THE ERROR.
5312      *
5313      *
5314      *****
5314 027646 000004 TST30: SCOPE
5315 027650 012737 000020 001676 MOV #20,$TIMES ;DO 20 ITERATIONS
5316      000031 IIA-$TN
5317      ;SET THE SKAD REGISTER
5318 027656 012737 030320 054230 MOV #TST31,SKAD ;IN CASE THE TEST ABORTS.
5319
5320 027664 113737 001502 001626 MOVB $TSTNM,$TMP0
5321 027672 012737 054076 000114 MOV #SPUR,@#CACHVEC
5322
5323 027700 005000 CLR R0 ;THIS IS THE COUNTER CONTAINING
5324      ;THE TEST DATA PATTERN
5325 027702 012737 027702 001512 IIA1: MOV #IIA1,$LPERR
5326 027710 004737 054470 JSR PC,PARCNT ;SET IF THIS TEST PATTERN HAS
5327 027714 032702 000001 BIT #BIT0,R2 ;THE PARITY BIT SET (1), IF NOT
5328 027720 001402 BEQ IIA2 ;GO TO THE NEXT PATTERN
5329 027722 000137 030300 JMP IIA7
5330 027726 012737 000030 177746 IIA2: MOV #SOM1,@#CONTRL ;SELECT GROUP ZERO.
5331 027734 012737 030204 000114 MOV #IIAR1,@#CACHVEC ;SET UP FOR THE ERROR
5332 027742 012705 030202 MOV #IIAT1,R5 ;MAKE THE TEST ADDRESS A
5333 027746 005715 TST (R5) ;HIT IN GROUP ZERO
5334 027750 005715 TST (R5) ;MAKE SURE IT IS A HIT
5335
5336      ;SEE IF REFERENCE ADDRESS
5337 027752 032737 000010 177752 BIT #10,@#HITMIS ;IS A HIT.
5338 027760 001007 BNE 1$
5339      ;IF NOT ERROR!
5340 027762 010537 001632 MOV R5,$TMP2
5341 027766 012737 000000 001630 MOV #0,$TMP1
5342 027774 104001 ERROR 1
5343
5344 027776 104420 SKIPT ;ERROR FATAL. GO TO NEXT TEST.
5345
5346
5347 030000 012704 000020 1$: MOV #20,R4 ;THIS PATTERN WILL BE
5348 030004 012702 177750 MOV #MAINT,R2 ;PUT IN THE MAINTENANCE
5349 030010 005001 CLR R1 ;REGISTER
5350 030012 010015 MOV R0,(R5) ;PUT THE TEST PATTERN IN
5351      ;THE TEST ADDRESS
5352 030014 000401 BR 64$
5353
5354      LOC=. ;GET THE PC TO AN EVEN WORD BOUNDARY...
5355      LOC--4&LOC
5356      LOC-LOC+4
5357      .=LOC
    
```

```

5358
5359
5360
5361
5362
5363 030020 010412
5364 030022 021500
5365 030024 010112
5366
5367 030026
5368
5369 030026 010037 001632
5370 030032 012737 030202 001634
5371 030040 005037 001636
5372 030044 104144
5373
5374 030046 012737 030244 000114
5375 030054 012737 030046 001512
5376 030062 012737 000044 177746
5377
5378 030070 012705 030202
5379 030074 005715
5380 030076 005715
5381
5382
5383 030100 032737 000010 177752
5384 030106 001007
5385
5386 030110 010537 001632
5387 030114 012737 000001 001630
5388 030122 104001
5389
5390 030124 104420
5391
5392
5393 030126 012704 000100
5394 030132 012702 177750
5395 030136 005001
5396 030140 010015
5397
5398 030142 000402
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408 030150 000240
5409 030152 010412
5410 030154 021500
5411 030156 010112
5412
5413 030160
    
```

```

64$: MOV R4,(R2)
    CMP (R5),R0
    MOV R1,(R2)
    ;THE REFERENCE TO THIS NEXT INSTRUCTION
    ;WILL MAKE THE COMPARE INSTRUCTION A HIT
    ;SO THAT NO SPURIOUS ERROR SHOULD OCCUR
    ;WHILE THE MAINTENANCE REGISTER IS SET!
    ;TURN ON THE MAINT. REG.
    ;THE REFERENCE TO (R5)
    ;SHOULD CAUSE THE ERROR.

IIA3:
    MOV R0,$TMP2
    MOV #IIAT1,$TMP3
    CLR $TMP4
    64$: ERROR 144
    ;THE ERROR DIDN'T OCCUR.
    ;REPORT FAILURE

IIA4: MOV #IIAR2,@#CACHVEC ;SET UP FOR THE GROUP ONE
    MOV #IIA4,$LPERR ;ERROR
    MOV #S1M0,@#CONTRL ;SELECT GROUP ONE

    MOV #IIAT1,R5 ;MAKE THE TEST ADDRESS A
    TST (R5) ;HIT, IN GROUP ONE.
    TST (R5)

    BIT #10,@#HITMIS ;SEE IF REFERENCE ADDRESS
    BNE 1$ ;IS A HIT.
    ;IF NOT ERROR.

    MOV R5,$TMP2
    MOV #1,$TMP1
    ERROR 1

    SKIP1 ;ERROR FATAL. GO TO NEXT TEST.

1$: MOV #100,R4 ;THIS PATTERN WILL BE
    MOV #MAINT,R2 ;PUT IN THE MAINT. REG.
    CLR R1
    MOV R0,(R5) ;PUT THE TEST PATTERN IN (R5),
    ;IIAT1.
    BR 50$ ;PUT THE NEXT INSTRUCTION EXECUTED
    ;ON AN EVEN WORD BOUNDARY SO THE
    ;SUBSEQUENT INSTRUCTION, A CMP,
    ;WILL BE A HIT.

    LOC=. ;GET THE PC TO AN EVEN WORD BOUNDARY...
    LOC=-4&LOC
    LOC=LOC+4
    .=LOC

50$: NOP ;FOR SCOPING WITH AN OSCILLOSCOPE.
    MOV R4,(R2) ;TURN ON THE MAINT. REG.
    CMP (R5),R0 ;THIS REFERENCE TO (R5) SHOULD
    MOV R1,(R2) ;CAUSE THE ERROR.

IIA5:
    
```

```

5414                                     ;THE ERROR DIDN'T OCCUR!
5415 030160 010037 001632                MOV    R0,$TMP2                ;REPORT FAILURE
5416 030164 012737 030202 001634        MOV    #IIAT1,$TMP3
5417 030172 005037 001636                CLR    $TMP4
5418 030176 104145                        64$:  ERROR    145
5419
5420 030200 000437                        IIA6:  BR      IIA7
5421
5422 030202 000000                        IIAT1:.WORD  0
5423
5424 030204                                IIAR1:
5425 030204 022737 004500 177744        CMP    #4500,@MEMERR ;MAKE SURE THE ERROR
5426 030212 001402                        BEQ    2$                ;REGISTER IS SET PROPERLY
5427 030214 000137 054076                1$:  JMP    SPUR
5428 030220 022737 030202 177740        2$:  CMP    #IIAT1,@LOADRS ;MAKE SURE THE ERROR
5429 030226 001372                        BNE    1$                ;OCCURRED AT THE CORRECT
5430                                     ;ADDRESS.
5431 030230 022626                        CMP    (SP)+,(SP)+        ;RESET THE STACK
5432 030232 012737 177777 177744        MOV    #-1,@MEMERR        ;CLEAR THE ERROR REGISTERS.
5433 030240 000137 030046                        JMP    IIA4                ;GO TEST GROUP ONE
5434 030244                                IIAR2:
5435 030244 022737 004600 177744        CMP    #4600,@MEMERR ;MAKE SURE THE ERROR
5436 030252 001402                        BEQ    2$                ;REGISTER IS SET PROPERLY
5437 030254 000137 054076                1$:  JMP    SPUR
5438 030260 022737 030202 177740        2$:  CMP    #IIAT1,@LOADRS ;MAKE SURE THE ERROR
5439 030266 001372                        BNE    1$                ;OCCURRED AT THE CORRECT
5440                                     ;ADDRESS.
5441 030270 022626                        CMP    (SP)+,(SP)+        ;RESET THE STACK
5442 030272 012737 177777 177744        MOV    #-1,@MEMERR        ;CLEAR THE ERROR REGISTERS.
5443
5444 030300 022700 000377                        IIA7:  CMP    #377,R0        ;INCREMENT THE TEST
5445 030304 001404                        BEQ    IIA8                ;PATTERN
5446 030306 062700 000001                        ADD    #1,R0
5447 030312 000137 027702                        JMP    IIA1
5448
5449 030316 104416                        IIA8:  RSET
5450
5451                                     ;:*****
5452                                     ;*TEST 31          CACHE DATA MEMORY PARITY CHECKERS HIGH BYTE TEST
5453                                     ;*
5454                                     ;*THIS IS A TEST OF THE TWO CACHE DATA MEMORY PARITY
5455                                     ;*CHECKERS FOR THE HIGH BYTE, ONE FOR EACH GROUP. THE
5456                                     ;*MAINTENANCE REGISTER ISUSED TO FORCE A PARITY A
5457                                     ;*PARITY ERROR AT EVERY DATA PATTERN WHICH HAS A ONE
5458                                     ;*PARITY BIT. NOTE THAT THE CACHE DATA MEMORY PARITY HAS,
5459                                     ;*EFFECTIVELY, ODD PARITY. THE MAINTENANCE FUNCTION ON THE
5460                                     ;*CACHE DATA MEMORY PARITY CHECKERS HAS THE EFFECT OF
5461                                     ;*FORCING THE PARITY BIT OF THE BYTE BEING CHECKED TO
5462                                     ;*ZERO. THIS MEANS THAT ONCE THIS MAINTENANCE FUNCTION
5463                                     ;*IS ENABLED THE ERROR WILL OCCUR ON A SUBSEQUENT
5464                                     ;*READ OF A BYTE WITH A ONE PARITY BIT, THAT IS
5465                                     ;*BYTES WITH ZERO PARITY BITS WILL NOT CAUSE THE ERROR.
5466                                     ;*
5467                                     ;:*****
5468 030320 000004                                TST31:  SCOPE
5469 030322 012737 000020 001676        MOV    #20,$TIMES        ;;DO 20 ITERATIONS
    
```

```

5470          000032          IIB=$TN
5471
5472 030330 012737 030774 054230          MOV      #TST32,SKAD          ;SET THE SKAD REGISTER
5473                                     ;IN CASE THE TEST ABORTS.
5474 030336 113737 001502 001626          MOV      $TSTNM,$TMP0
5475 030344 012737 054076 000114          MOV      #SPUR,@#CACHVEC
5476
5477 030352 005000          CLR      R0          ;THIS IS THE COUNTER CONTAINING
5478                                     ;THE TEST DATA PATTERN
5479 030354 012737 030354 001512 IIB1:  MOV      #IIB1,$LPERP
5480 030362 004737 054470          JSR      PC,PARCNT          ;SET IF THIS TEST PATTERN HAS
5481 030366 032702 000001          BIT      #BIT0,R2          ;THE PARITY BIT SET (1), IF NOT
5482 030372 001402          BEQ     IIB2          ;GO TO THE NEXT PATTERN
5483 030374 000137 030754          JMP     IIB7
5484 030400 012737 000030 177746 IIB2:  MOV      #SOM1,@#CONTRL          ;SELECT GROUP ZERO.
5485 030406 012737 030660 000114          MOV      #IIBR1,@#CACHVEC          ;SET UP FOR THE ERROR
5486 030414 012705 030656          MOV      #IIBT1,R5          ;MAKE THE TEST ADDRESS A
5487 030420 005715          TST     (R5)          ;HIT IN GROUP ZERO
5488 030422 005715          TST     (R5)          ;MAKE SURE IT IS A HIT
5489
5490
5491 030424 032737 000010 177752          BIT      #10,@#HITMIS          ;SEE IF REFERENCE ADDRESS
5492 030432 001007          BNE     1$          ;IS A HIT.
5493                                     ;IF NOT ERROR.
5494 030434 010537 001632          MOV      R5,$TMP2
5495 030440 012737 000000 001630          MOV      #0,$TMP1
5496 030446 104001          ERROR  1
5497
5498 030450 104420          SKIPT          ;ERROR FATAL. GO TO NEXT TEST.
5499
5500
5501 030452 012704 000040          1$:  MOV      #40,R4          ;THIS PATTERN WILL BE
5502 030456 012702 177750          MOV      #MAINT,R2          ;PUT IN THE MAINTENANCE
5503 030462 005001          CLR      R1          ;REGISTER
5504 030464 010015          MOV      R0,(R5)          ;PUT THE TEST PATTERN IN
5505                                     ;THE TEST ADDRESS
5506 030466 000402          BR      64$
5507
5508          030470          LOC-          ;GET THE PC TO AN EVEN WORD BOUNDARY!!
5509          030470          LOC=-4&LOC
5510          030474          LOC=LOC+4
5511          030474          .=LOC
5512
5513                                     ;THE REFERENCE TO THIS NEXT INSTRUCTION
5514                                     ;WILL MAKE THE COMPARE INSTRUCTION A HIT
5515                                     ;SO THAT NO SPURIOUS ERROR SHOULD OCCUR
5516                                     ;WHILE THE MAINTENANCE REGISTER IS SET!
5517 030474 010412          64$:  MOV      R4,(R2)          ;TURN ON THE MAINT. REG.
5518 030476 021500          CMP      (R5),R0          ;THE REFERENCE TO (R5)
5519 030500 010112          MOV      R1,(R2)          ;SHOULD CAUSE THE ERROR.
5520
5521 030502          IIB3:
5522                                     ;THE ERROR DIDN'T OCCUR!
5523 030502 010037 001632          MOV      R0,$TMP2          ;REPORT FAILURE
5524 030506 012737 030656 001634          MOV      #IIBT1,$TMP3
5525 030514 005037 001636          CLR      $TMP4
    
```

```

C 11
CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 107
CEKBD.D.P11 16-MAY-79 08:58 T31 CACHE DATA MEMORY PARITY CHECKERS HIGH BYTE TEST SEQ 0132

5526 030520 104146 64$: ERROR 146
5527
5528 030522 012737 030720 000114 I1B4: MOV #I1BR2,@#CACHVEC ;SET UP FOR THE GROUP ONE
5529 030530 012737 030522 001512 MOV #I1B4,$LPERR ;ERROR
5530 030536 012737 000044 177746 MOV #S1M0,@#CONTRL ;SELECT GROUP ONE
5531
5532 030544 012705 030656 MOV #I1BT1,R5 ;MAKE THE TEST ADDRESS A
5533 030550 005715 TST (R5) ;HIT, IN GROUP ONE.
5534 030552 005715 TST (R5)
5535
5536 ;SEE IF REFERENCE ADDRESS
5537 030554 032737 000010 177752 BIT #10,@#HITMIS ;IS A HIT.
5538 030562 001007 BNE 1$
5539 ;IF NOT ERROR!
5540 030564 010537 001632 MOV R5,$TMP2
5541 030570 012737 000001 001630 MOV #1,$TMP1
5542 030576 104001 ERROR 1
5543
5544 030600 104420 SKIPT ;ERROR FATAL. GO TO NEXT TEST.
5545
5546
5547 030602 012704 000200 1$: MOV #200,R4 ;THIS PATTERN WILL BE
5548 030606 012702 177750 MOV #MAINT,R2 ;PUT IN THE MAINT. REG.
5549 030612 005001 CLR R1
5550 030614 010015 MOV R0,(R5) ;PUT THE TEST PATTERN IN (R5),
5551 ;I1BT1.
5552 030616 000402 BR 50$ ;PUT THE NEXT INSTRUCTION EXECUTED
5553 ;ON AN EVEN WORD BOUNDARY SO THE
5554 ;SUBSEQUENT INSTRUCTION, A CMP,
5555 ;WILL BE A HIT.
5556
5557 030620 LOC=. ;GET THE PC TO AN EVEN WORD BOUNDARY..
5558 030620 LOC--4&LOC
5559 030624 LOC=LOC+4
5560 030624 .=LOC
5561
5562 030624 000240 50$: NOP ;FOR SCOPING WITH AN OSCILLOSCOPE.
5563 030626 010412 MOV R4,(R2) ;TURN ON THE MAINT. REG.
5564 030630 021500 CMP (R5),R0 ;THIS REFERENCE TO (R5) SHOULD
5565 030632 010112 MOV R1,(R2) ;CAUSE THE ERROR.
5566
5567 030634 I1B5: ;THE ERROR DIDN'T OCCUR.
5568 ;REPORT FAILURE
5569 030634 010037 001632 MOV R0,$TMP2
5570 030640 012737 030656 001634 MOV #I1BT1,$TMP3
5571 030646 005037 001636 CLR $TMP4
5572 030652 104147 64$: ERPOR 147
5573
5574 030654 000437 I1B6: BR I1B7
5575
5576 030656 000000 I1B7: .WORD 0
5577
5578 030660 I1BR1:
5579 030660 022737 004500 177744 CMP #4500,@#MEMERR ;MAKE SURE THE ERROR
5580 030666 001402 BEQ 2$ ;REGISTER IS SET PROPERLY
5581 030670 000137 054076 1$: JMP SPUR

```



```

5638
5639 031054 012737 000001 031520      MOV      #1,QQFLG1      ;RESPECTIVELY.
5640                                     ;QQFLG ONE CONTAINS THE
5641                                     ;NUMBER OF THE GROUP
5642                                     ;BEING TESTED, INITIALLY 1.
5643 031062 012703 031062      QQ2:    MOV      #QQ2,R3      ;MAKE LOCATIONS QQ1
5644 031066 012704 001000      MOV      #1000,R4      ;THROUGH QQ2 + 2000 (OCT)
5645 031066 013737 031534 177746 1$:    MOV      QQGM,@#CONTRL ;HITS IN THE GROUP NOT
5646 031100 005713                                     (R3)      ;BEING TESTED WHILE
5647 031102 013737 031532 177746      MOV      QQGS,@#CONTRL ;GETTING THESE LOCATIONS
5648 031110 005763 002000      TST     2000(R3)      ;TO BE MISSES IN THE
5649 031114 062703 000002      ADD     #2,R3        ;GROUP THAT IS BEING
5650 031120 077414      SOB     R4,1$        ;TESTED
5651 031122 012704 001000      MOV      #1000,R4      ;MAKE LOCATIONS TESTR2
5652 031126 012705 142000      MOV      #TESTR2,R5    ;THROUGH TESTR2+2000(OCT)
5653 031132 013703 031532      MOV      QQGS,R3      ;HITS IN THE GROUP
5654 031136 042703 177717      BIC     #177717,R3    ;BEING TESTED WHILE
5655 031142 010337 177746      MOV      R3,@#CONTRL  ;WRITING THE BACKGROUND
5656 031146 013715 031522      QQ3:    MOV      QQPAT1,(R5) ;PATTERN, IN QQPAT1, IN
5657 031152 005715      TST     (R5)
5658 031154 005725      TST     (R5)+        ;THEM. MAKE SURE THEY
5659 031156 032737 000010 177752      BIT     #10,@#HITMIS ;ARE HITS
5660 031164 001011      BNE     QQ4
5661 031166 013737 031520 001630      MOV      QQFLG1,$TMP1 ;IF NOT ERROR
5662 031174 010537 001632      MOV      R5,$TMP2
5663 031200 062737 177776 001632      ADD     #-2,$TMP2
5664 031206 104001      1$:    ERRGR 1
5665 031210 077422      QQ4:    SOB     R4,QQ3
5666 031212 013703 031534      MOV      QQGM,R3      ;FROM NOW ON SELECT
5667 031216 042703 177717      BIC     #177717,R3    ;THE GROUP NOT BEING
5668 031222 010337 177746      MOV      R3,@#CONTRL ;TESTED
5669
5670 031226 012704 031536      MOV      #QQ10,R4     ;THE THREE ROUTINES
5671 031232 042704 176000      BIC     #176000,R4    ;QQ10-QQ11, QQ12-QQ13 AND
5672 031236 012705 031612      MOV      #QQ11,R5     ;QQ14-QQ15 ARE IDENTICAL
5673 031242 042705 176000      BIC     #176000,R5    ;EXCEPT FOR WHAT PART
5674 031246 020405      CMP     R4,R5        ;OF THE CACHE GROUP THAT
5675 031250 002407      BLT     QQ5          ;IS NOT BEING TEST THEY
5676 031252 012737 031614 031512      MOV      #QQ12,QQLO   ;LIE IN. HERE DECIDE
5677 031260 012737 031672 031514      MOV      #QQ14,QQHI   ;WHICH TWO OF THE
5678 031266 000450      BR      QQ8          ;ABOVE THREE IS APPROPRIATE
5679 031270 012704 031614      QQ5:    MOV      #QQ12,R4     ;FOR THIS TEST.
5680 031274 042704 176000      BIC     #176000,R4
5681 031300 012705 031670      MOV      #QQ13,R5
5682 031304 042705 176000      BIC     #176000,R5
5683 031310 020405      CMP     R4,R5
5684 031312 002407      BLT     QQ6
5685 031314 013737 031672 031512      MOV      QQ14,QQLO
5686 031322 013737 031536 031514      MOV      QQ10,QQHI
5687 031330 000427      BR      QQ8
5688 031332 013704 031536      QQ6:    MOV      QQ10,R4
5689 031336 042704 176000      BIC     #176000,R4
5690 031342 012705 031614      MOV      #QQ12,R5
5691 031346 042705 176000      BIC     #176000,R5
5692 031352 020405      CMP     R4,R5
5693 031354 003007      BGT     QQ7
    
```

```

5694 031356 012737 031536 031512      MOV      #QQ10,QQLO
5695 031364 012737 031614 031514      MOV      #QQ12,QQHI
5696 031372 000406                      BR       QQ8
5697 031374 012737 031614 031512  QQ7:   MOV      #QQ12,QQLO
5698 031402 012737 031536 031514      MOV      #QQ10,QQHI
5699
5700 031410 012702 142000                QQ8:   MOV      #TESTR2,R2      ;INITIALIZE FOR EITHER
5701 031414 012701 140000                MOV      #TESTR1,R1      ;THE GALLOPING ONES OR
5702 031420 012705 001000                MOV      #1000,R5        ;GALLOPING ZEROES TEST
5703                                     ;WHICH IS PENDING.
5704 031424 012737 032034 000114      MOV      #QQERR1,@#CACHVEC ;IF THE TEST FAILS A
5705                                     ;PARITY ABORT IS LIKELY
5706                                     ;SO SET UP TO GO THE
5707                                     ;ERROR ROUTINE.
5708 031432 012737 031440 001512      MOV      #QQ9,$LPERR     ;SET THE LOOP ERROR
5709                                     ;ADDRESS FOR THE BEGINNING
5710                                     ;OF THE PASS ROUTINE.
5711
5712 031440 012703 142000                QQ9:   MOV      #TESTR2,R3      ;THIS DOES ONE PASS OF
5713 031444 012704 001000                MOV      #1000,R4        ;THE TEST FOR EACH LOCATION.
5714 031450 005112                      COM      (R2)            ;PUT THE GALLOPING PATTERN
5715                                     ;IN THE MEMORY.
5716
5717 031452 010100                QQ9.5: MOV      R1,R0          ;SEE WHICH OF THE
5718 031454 042700 176000                BIC      #176000,R0      ;TWO ROUTINES (QQ10,QQ12 OR
5719 031460 013737 031514 031524      MOV      QQHI,QQTMP1     ;QQ14) SHOULD FINISH
5720 031466 042737 176000 031524      BIC      #176000,QQTMP1  ;SETTING FOR THIS TEST
5721 031474 020037 031524                CMP      R0,QQTMP1      ;PASS.
5722 031500 002402                      BLT     1$
5723 031502 000177 000004                JMP     @QQLO
5724 031506 000177 000002                1$:    JMP     @QQHI
5725
5726 031512 000000                QQLO:  .WORD  0           ;QQLO AND QQHI CONTAIN THE
5727 031514 000000                QQHI:  .WORD  0           ;ADDRESSES OF THE ROUTINES
5728                                     ;TO BE USED IN SETTING UP
5729                                     ;FOR A PASS.
5730 031516 000000                QQFLG2: .WORD  0         ;1 IF DOING GALLOPING 1'S TEST.
5731                                     ;0 IF DOING GALLOPING 0'S TEST.
5732 031520 000000                QQFLG1: .WORD  0         ;GROUP BEING TESTED, 1 OR 0.
5733 031522 000000                QQPAT1: .WORD  0         ;0 OR 1 BACKGROUND PATTERN.
5734 031524 000000                QQTMP1: .WORD  0         ;USED AS TEMPORARY STORAGE.
5735 031526 000000                QQTMP2: .WORD  0
5736 031530 000000                QQTMP3: .WORD  0
5737 031532 000000                QQGS:  .WORD  0           ;THESE REGISTERS HOLD PATTERNS
5738 031534 000000                QQGM:  .WORD  0           ;WHICH ARE TO BE LOADED INTO THE
5739                                     ;CACHE CONTROL REGISTER.
5740
5741                                     ;THIS ROUTINE IS USED TO SET UP THE INSTRUCTIONS:
5742                                     ;
5743                                     ;
5744                                     ;
5745                                     ;IN POSITION, AS HITS IN THE GROUP NOT BEING TESTED.
5746 031536 000240                QQ10:  NOP              ;USED AS A BUFFER SO
5747 031540 000240                NOP              ;THIS CODE WON'T WIPE
5748                                     ;OUT DESIRED HITS
5749 031542 012711 022312      MOV      #022312,(R1)    ;020323=(CMP (R3)+,(R2)

```



```

5750 031546 005711          TST      (R1)
5751 031550 012761 077402 000002  MOV     #077402,2(R1)  ;077402=(SOB R4,..-2)
5752 031556 005761 000002          TST     2(R1)
5753 031562 012761 000137 000004  MOV     #000137,4(R1)  ;000137=(JMP @#0016)
5754 031570 005761 000004          TST     4(R1)          ;0016
5755 031574 012761 031750 000006  MOV     #0016,6(R1)
5756 031602 005761 000006          TST     6(R1)
5757 031606 000111          JMP     (R1)          ;GO DO A PASS.
5758 031610 000240          NOP
5759 031612 000240  Q011:  NOP
5760
5761          ;THIS ROUTINE IS USED TO SET UP THE INSTRUCTIONS:
5762          1$:  CMP     (R3)+,(R2)
5763          :
5764          :      SOB     R4,1$
5765          :      JMP     @#0016
5766          ;IN POSITION, AS HITS IN THE GROUP NOT BEING TESTED.
5766 031614 000240  Q012:  NOP          ;USED AS A BUFFER SO
5767 031616 000240          NOP          ;THIS CODE WON'T WIPE
5768          :      OUT    DESIRED HITS
5769 031620 012711 022312          MOV     #022312,(R1)  ;020323=(CMP (R3)+,(R2)
5770 031624 005711          TST     (R1)
5771 031626 012761 077402 000002  MOV     #077402,2(R1)  ;077402=(SOB R4,..-2)
5772 031634 005761 000002          TST     2(R1)
5773 031640 012761 000137 000004  MOV     #000137,4(R1)  ;000137=(JMP @#0016)
5774 031646 005761 000004          TST     4(R1)          ;0016
5775 031652 012761 031750 000006  MOV     #0016,6(R1)
5776 031660 005761 000006          TST     6(R1)
5777 031664 000111          JMP     (R1)          ;GO DO A PASS.
5778 031666 000240          NOP
5779 031670 000240  Q013:  NOP
5780
5781          ;THIS ROUTINE IS USED TO SET UP THE INSTRUCTIONS:
5782          1$:  CMP     (R3)+,(R2)
5783          :
5784          :      SOB     R4,1$
5785          :      JMP     @#0016
5786          ;IN POSITION, AS HITS IN THE GROUP NOT BEING TESTED.
5786 031672 000240  Q014:  NOP          ;USED AS A BUFFER SO
5787 031674 000240          NOP          ;THIS CODE WON'T WIPE
5788          :      OUT    DESIRED HITS
5789 031676 012711 022312          MOV     #022312,(R1)  ;020323=(CMP (R3)+,(R2)
5790 031702 005711          TST     (R1)
5791 031704 012761 077402 000002  MOV     #077402,2(R1)  ;077402=(SOB R4,..-2)
5792 031712 005761 000002          TST     2(R1)
5793 031716 012761 000137 000004  MOV     #000137,4(R1)  ;000137=(JMP @#0016)
5794 031724 005761 000004          TST     4(R1)          ;0016
5795 031730 012761 031750 000006  MOV     #0016,6(R1)
5796 031736 005761 000006          TST     6(R1)
5797 031742 000111          JMP     (R1)          ;GO DO A PASS.
5798 031744 000240          NOP
5799 031746 000240  Q015:  NOP
5800
5801 031750 005122  Q016:  COM     (R2)+      ;PASS DONE. RESTORE THE
5802          :      BACKGND PATTERN.
5803
5804 031752 062701 000002  Q017:  ADD     #2,R1        ;GO TO NEXT LOCATION FOR
5805          :      NEXT PASS.

```

```

5806 031756 005305          DEC      R5          ;DO ANOTHER PASS?
5807 031760 001402          BEQ      1$
5808 031762 000137 031440    JMP      QQ9
5809 031766                1$:
5810 031766 012737 000044 031534    MOV      #S1M0,QQGM   ;TESTED GROUP 1 NOW GO BACK
5811 031774 012737 000030 031532    MOV      #S0M1,QQGS   ;AND TEST GROUP 0
5812 032002 005337 031520          DEC      QQFLG1
5813 032006 001002          BNE      QQ18
5814 032010 000137 031062          JMP      QQ2
5815
5816 032014 012737 177777 031522  QQ18:  MOV      #-1,QQPAT1   ;GALLOPING 1'S TEST IS
5817 032022 005337 031516          DEC      QQFLG2       ;COMPLETE, ON BOTH GROUPS,
5818 032026 001041          BNE      QQ19         ;SET THE BACKGROUND PATTERN
5819 032030 000137 031032          JMP      QQ1          ;FOR GALLOPING 0'S AND GO
5820                                     ;BACK TO PERFORM THIS TEST
5821                                     ;ON BOTH GROUPS.
5822
5823 032034 013737 177744 001630  QQERR1: MOV      @MEMERR,$TMP1 ;COME HERE IF DURING THE
5824 032042 013737 177740 001632          MOV      @LOADRS,$TMP2 ;TEST A TRAP OR ABORT
5825 032050 013737 177742 001634          MOV      @HIADRS,$TMP3 ;OCCURRED TO CACHVEC
5826 032056 011637 001636          MOV      (SP),$TMP4
5827 032062 022626          CMP      (SP)+,(SP)+
5828 032064 010137 001640          MOV      R1,$TMP5
5829 032070 013737 031520 001642          MOV      QQFLG1,$TMP6
5830 032076 032737 000600 001630          BIT      #600,$TMP1
5831 032104 001002          BNE      QQERR2
5832 032106 104002          ERROR   2
5833 032110 000406          BR      QQERR4
5834 032112 005737 031522          QQERR2: TST      QQPAT1 ;GALLOPING 1' OR 0'S?
5835 032116 001002          BNE      QQERR3
5836 032120 104003          ERROR   3           ;0'S.
5837 032122 000401          BR      QQERR4
5838 032124 104004          QQERR3: ERROR   4           ;1'S
5839 032126 000137 031750          QQERR4: JMP      QQ16     ;CONTINUE?
5840
5841 032132          QQ19:          ;DONE! PERHAPS PRINT SUMMARY.
5842                                     ;?????
5843
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
    
```

```

:*****
:*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.
    
```

```

5862      ;*FOR EACH PERMUTATION OF THE TEST PATTERNS THIS ROUTINE FIRST WRITES
5863      ;*'UP' (SET 0 FIRST THEN 1,2 AND 3) THEN 'DOWN' (SET 3 FIRST THEN 2,1 AND 0).
5864      ;*
5865      ;*****
5866 032132 00J004      TST33: SCOPE
5867 032134 012737 000040 001676      MOV      #40,$TIMES      ;;DO 40 ITERATIONS
5868      ;SET THE SKAD REGISTER
5869 032142 012737 033676 054230      MOV      #TST34,SKAD    ;IN CASE THE TEST ABORTS.
5870
5871
5872 032150 113737 001502 001626      MOVB     $TSTNM,$TMP0    ;PUT THE TEST NUMBER IN
5873      ;$TMP0 FOR PRINT OUT.
5874 032156 012737 054076 000114      MOV      #SPUR,@#CACHVEC ;EXPECT NO PARITY ERRORS.
5875
5876 032164 012737 000014 177746      KK1:    MOV      #MOM1,@#CONTRL ;FORCE MISSES AND
5877 032172 005037 033532      CLR      KKPAT1          ;INITIALIZE THE TEST PATTERN
5878 032176 012737 177777 033534      MOV      #177777,KKPAT2 ;TABLE
5879 032204 012737 125252 033536      MOV      #125252,KKPAT3
5880 032212 012737 052525 033540      MOV      #52525,KKPAT4
5881
5882 032220 005037 033526      CLR      KKFLG1          ;INITIALIZE KKFLG1:
5883      ;0 MEANS WRITE PATTERNS IN
5884      ;IN THE UPWARD DIRECTION
5885      ;1 MEANS WRITE PATTERNS IN
5886      ;THE DOWNWARD DIRECTION
5887
5888 032224 012700 033546      KK2:    MOV      #KKTMP2,R0    ;ESTABLISH AN OFFSET FOR
5889 032230 042700 176003      BIC      #176003,R0      ;A TEST AREA WHOSE HITS
5890      ;WILL NOT BE INTERFERRED WITH BY
5891 032234 010001      MOV      R0,R1          ;THE CYCLES CAUSED WHILE
5892 032236 062701 140000      ADD      #TESTR1,R1      ;FETCHING THE TEST CODE.
5893 032242 010002      MOV      R0,R2
5894 032244 062702 142000      ADD      #TESTR2,R2
5895
5896 032250 010137 001640      MOV      R1,$TMP5        ;SAVE THE ADDRESSES OF
5897 032254 010137 001642      MOV      R1,$TMP6        ;THE FOUR TEST WORD LOCATIONS,
5898 032260 062737 000002 001642      ADD      #2,$TMP6        ;FOR TYPE OUT IN CASE
5899 032266 010237 001644      MOV      R2,$TMP7        ;OF ERROR.
5900 032272 010237 001646      MOV      R2,$TMP10
5901 032276 062737 000002 001646      ADD      #2,$TMP10
5902
5903 032304 012705 033534      MOV      #KKPAT2,R5      ;A POINTER USED IN GENERATING
5904      ;EVERY PERMUTATION OF THE TEST
5905      ;PATTERNS.
5906 032310 012700 000006      MOV      #6,R0           ;R0 AND KKCNT1 ARE ALSO USED
5907 032314 012737 000004 033530      MOV      #4,KKCNT1      ;IN GENERATING THE PERMUTATIONS.
5908
5909 032322 012737 032330 001512      KK3:    MOV      #KK3,$LPERR    ;WHEN LOOPING ON ERROR GO TO KK3.
5910 032330 000240      NOP                       ;FOR SCOPING PER POSES
5911 032332 012737 000034 177746      MOV      #SOMOM1,@#CONTRL ;MAKE THE TEST AREA HITS
5912 032340 005711      TST      (R1)            ;IN THE CACHE GROUPS.
5913 032342 005761 000002      TST      2(R1)
5914 032346 012737 000054 177746      MOV      #S1MOM1,@#CONTRL
5915 032354 005712      TST      (R2)
5916 032356 005762 000002      TST      2(R2)
5917 032362 005037 177746      CLR      @#CONTRL
    
```

```

5918
5919
5920 032366 005711          TST      (R1)
5921
5922
5923 032370 032737 000010 177752  BIT      #10,@WHITMIS ;SEE IF REFERENCE ADDRESS
5924 032376 001006          BNE      1$           ;IS A HIT.
5925
5926 032400 010137 001632          MOV      R1,$TMP2    ;IF NOT ERROR!
5927 032404 012737 000000 001630  MOV      #0,$TMP1
5928 032412 104001          ERROR    1
5929
5930
5931
5932 032414          1$:
5933
5934 032414 005761 000002          TST      2(R1)
5935
5936
5937 032420 032737 000010 177752  BIT      #10,@WHITMIS ;SEE IF REFERENCE ADDRESS
5938 032426 001011          BNE      2$           ;IS A HIT.
5939
5940 032430 010137 001632          MOV      R1,$TMP2    ;IF NOT ERROR!
5941 032434 062737 000002 001632  ADD      #2,$TMP2
5942 032442 012737 000000 001630  MOV      #0,$TMP1
5943 032450 104001          ERROR    1
5944
5945
5946
5947 032452          2$:
5948
5949 032452 005712          TST      (R2)
5950
5951
5952 032454 032737 000010 177752  BIT      #10,@WHITMIS ;SEE IF REFERENCE ADDRESS
5953 032462 001006          BNE      3$           ;IS A HIT.
5954
5955 032464 010237 001632          MOV      R2,$TMP2    ;IF NOT ERROR!
5956 032470 012737 000001 001630  MOV      #1,$TMP1
5957 032476 104001          ERROR    1
5958
5959
5960
5961 032500          3$:
5962
5963 032500 005762 000002          TST      2(R2)
5964
5965
5966 032504 032737 000010 177752  BIT      #10,@WHITMIS ;SEE IF REFERENCE ADDRESS
5967 032512 001011          BNE      4$           ;IS A HIT.
5968
5969 032514 010237 001632          MOV      R2,$TMP2    ;IF NOT ERROR!
5970 032520 062737 000002 001632  ADD      #2,$TMP2
5971 032526 012737 000001 001630  MOV      #1,$TMP1
5972 032534 104001          ERROR    1
5973

```

K 11

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) 16-MAY-79 09:11 PAGE 115
 CEKBDD.P11 16-MAY-79 08:58 T33 CACHE DATA MEMORY CHIP SELECTION LOGIC TEST SEQ 0140

```

5974
5975
5976
5977 032536 005737 033526 4$: TST KKFLG1 ;SEE IF THE TST PATTERN
5978 ;SHOULD BE WRITTEN UPWARD
5979 ;OR DOWNWARD.
5980 032542 001045 BNE KK4 ;BRANCH IF DOWNWARD
5981 ;OTHERWISE WRITE IT IN THE
5982 ;UPWARD DIRECTION.
5983 032544 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
5984 032552 013703 033532 MOV KKPAT1,R3 ;LOCATION KKPAT1, INTO THE
5985 032556 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 0
5986 032562 010361 000000 MOV R3,0(R1)
5987 032566 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
5988 032574 013703 033534 MOV KKPAT2,R3 ;LOCATION KKPAT2, INTO THE
5989 032600 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 2
5990 032604 010361 000002 MOV R3,2(R1)
5991 032610 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
5992 032616 013703 033536 MOV KKPAT3,R3 ;LOCATION KKPAT3, INTO THE
5993 032622 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 0
5994 032626 010362 000000 MOV R3,0(R2)
5995 032632 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
5996 032640 013703 033540 MOV KKPAT4,R3 ;LOCATION KKPAT4, INTO THE
5997 032644 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 2
5998 032650 010362 000002 MOV R3,2(R2)
5999 032654 000444 BR KK5
6000 032656 KK4: ;WRITE THE PATTERN IN THE
6001 ;DOWNWARD DIRECTION
6002 032656 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6003 032664 013703 033540 MOV KKPAT4,R3 ;LOCATION KKPAT4, INTO THE
6004 032670 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 2
6005 032674 010362 000002 MOV R3,2(R2)
6006 032700 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6007 032706 013703 033536 MOV KKPAT3,R3 ;LOCATION KKPAT3, INTO THE
6008 032712 005037 177746 CLR @#CONTRL ;ADDRESS IN R2 PLUS 0
6009 032716 010362 000000 MOV R3,0(R2)
6010 032722 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6011 032730 013703 033534 MOV KKPAT2,R3 ;LOCATION KKPAT2, INTO THE
6012 032734 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 2
6013 032740 010361 000002 MOV R3,2(R1)
6014 032744 012737 000014 177746 MOV #MOM1,@#CONTRL ;WRITE THE TEST PATTERN, FROM
6015 032752 013703 033532 MOV KKPAT1,R3 ;LOCATION KKPAT1, INTO THE
6016 032756 005037 177746 CLR @#CONTRL ;ADDRESS IN R1 PLUS 0
6017 032762 010361 000000 MOV R3,0(R1)
6018
6019 032766 KK5:
6020 032766 012737 000014 177746 MOV #MOM1,@#CONTRL
6021 032774 013703 033532 MOV KKPAT1,R3 ;SEE IF THE TEST PATTERN WAS
6022 033000 005037 177746 CLR @#CONTRL ;WRITTEN OR IS READ CORRECTLY.
6023 033004 016104 000000 MOV 0(R1),R4
6024
6025 ;SEE IF REFERENCE ADDRESS
6026 033010 032737 000010 177752 BIT #10,@#HITMIS ;IS A HIT.
6027 033016 001006 BNE 64$
6028 ;IF NOT ERROR.
6029 033020 010137 001632 MOV R1,$TMP2

```

6030	033024	012737	000000	001630	MOV	#0,STMP1	
6031	033032	104001			ERROR	1	
6032							
6033							
6034	033034	020403			64\$: CMP	R4,R3	
6035	033036	001402			BEQ	KK6	
6036	033040	004737	033556		JSR	PC,KKERR1	
6037							
6038	033044				KK6: MOV	#MOM1,@CONTRL	
6039	033044	012737	000014	177746	MOV	KKPAT2,R3	
6040	033052	013703	033534		CLR	@CONTRL	;SEE IF THE TEST PATTERN WAS
6041	033056	005037	177746		MOV	2(R1),R4	;WRITTEN OR IS READ CORRECTLY.
6042	033062	016104	000002				
6043							
6044							
6045	033066	032737	000010	177752	BIT	#10,@WHITMIS	;SEE IF REFERENCE ADDRESS
6046	033074	001011			BNE	64\$;IS A HIT.
6047							;IF NOT ERROR.
6048	033076	010137	001632		MOV	R1,STMP2	
6049	033102	062737	000002	001632	ADD	#2,STMP2	
6050	033110	012737	000000	001630	MOV	#0,STMP1	
6051	033116	104001			ERROR	1	
6052							
6053							
6054	033120	020403			64\$: CMP	R4,R3	
6055	033122	001402			BEQ	KK7	
6056	033124	004737	033570		JSR	PC,KKERR2	
6057							
6058	033130				KK7: MOV	#MOM1,@CONTRL	
6059	033130	012737	000014	177746	MOV	KKPAT3,R3	
6060	033136	013703	033536		CLR	@CONTRL	;SEE IF THE TEST PATTERN WAS
6061	033142	005037	177746		MOV	0(R2),R4	;WRITTEN OR IS READ CORRECTLY.
6062	033146	016204	000000				
6063							
6064							
6065	033152	032737	000010	177752	BIT	#10,@WHITMIS	;SEE IF REFERENCE ADDRESS
6066	033160	001006			BNE	64\$;IS A HIT.
6067							;IF NOT ERROR.
6068	033162	010237	001632		MOV	R2,STMP2	
6069	033166	012737	000001	001630	MOV	#1,STMP1	
6070	033174	104001			ERROR	1	
6071							
6072							
6073	033176	020403			64\$: CMP	R4,R3	
6074	033200	001402			BEQ	KK8	
6075	033202	004737	033610		JSR	PC,KKERR3	
6076							
6077	033206				KK8: MOV	#MOM1,@CONTRL	
6078	033206	012737	000014	177746	MOV	KKPAT4,R3	
6079	033214	013703	033540		CLR	@CONTRL	;SEE IF THE TEST PATTERN WAS
6080	033220	005037	177746		MOV	2(R2),R4	;WRITTEN OR IS READ CORRECTLY.
6081	033224	016204	000002				
6082							
6083							
6084	033230	032737	000010	177752	BIT	#10,@WHITMIS	;SEE IF REFERENCE ADDRESS
6085	033236	001011			BNE	64\$;IS A HIT.

```

6086                                     ;IF NOT ERROR.
6087 033240 010237 001632             MOV    R2,$TMP2
6088 033244 062737 000002 001632     ADD    #2,$TMP2
6089 033252 012737 000001 001630     MOV    #1,$TMP1
6090 033260 104001                     ERROR  1
6091
6092
6093 033262 020403                     648:   CMP    R4,R3
6094 033264 001402                     BEQ
6095 033266 004737 033624             JSR    PC, KKERR4
6096
6097 033272 005737 033526             KK10:  TST    KKFLG1
6098 033276 001005                     BNE    KK11
6099 033300 012737 177777 033526     MOV    #-1, KKFLG1
6100 033306 000137 032330             JMP    KK3
6101
6102
6103 033312 005037 033526             KK11:  CLR    KKFLG1
6104 033316 012737 000014 177746     MOV    #MOM1, @#CONTRL
6105
6106 033324 020527 033540             CMP    R5, #KKPAT4
6107 033330 001011                     BNE
6108
6109 033332 011537 033542             MOV    (R5), KKPAT5
6110 033336 013715 033534             MOV    KKPAT2, (R5)
6111 033342 012705 033534             MOV    #KKPAT2, R5
6112 033346 013715 033542             MOV    KKPAT5, (R5)
6113 033352 000406                     BR     KK13
6114
6115 033354 012537 033542             KK12:  MOV    (R5)+, KKPAT5
6116 033360 011565 177776             MOV    (R5), -2(R5)
6117 033364 013715 033542             MOV    KKPAT5, (R5)
6118
6119 033370 005300                     KK13:  DEC    R0
6120 033372 001402                     BEQ    KK14
6121 033374 000137 032330             JMP    KK3
6122
6123 033400 012700 000006             KK14:  MOV    #6, R0
6124 033404 013737 033532 033542     MOV    KKPAT1, KKPAT5
6125 033412 005337 033530             DEC    KKCNT1
6126
6127 033416 022737 000003 033530     CMP    #3, KKCNT1
6128 033424 001010                     BNE    KK15
6129
6130 033426 013737 033534 033532     MOV    KKPAT2, KKPAT1
6131 033434 013737 033542 033534     MOV    KKPAT5, KKPAT2
6132 033442 000137 032330             JMP    KK3
6133
6134 033446 022737 000002 033530     KK15:  CMP    #2, KKCNT1
6135 033454 001010                     BNE    KK16
6136
6137 033456 013737 033536 033532     MOV    KKPAT3, KKPAT1
6138 033464 013737 033542 033536     MOV    KKPAT5, KKPAT3
6139 033472 000137 032330             JMP    KK3
6140
6141 033476 022737 000001 033530     KK16:  CMP    #1, KKCNT1
    
```

```

6142 033504 001073          BNE      KK17          ;BRANCH IF DONE.
6143
6144 033506 013737 033540 033532      MOV      KKPAT4,KKPAT1
6145 033514 013737 033542 033540      MOV      KKPAT5,KKPAT4
6146 033522 000137 032330          JMP      KK3          ;GO DO NEXT PERMUTATION.
6147
6148
6149 033526 000000          KKFLG1: .WORD 0          ;0 IF STORING PATTERN UPWARD
6150                                     ;1 IF STORING DOWNWARD.
6151
6152 033530 000000          KKCNT1: .WORD 0          ;COUNTER USED IN GENERATING
6153                                     ;THE TEST PATTERN PERMUTATIONS.
6154
6155 033532 000000          KKPAT1: .WORD 0          ;TEST PATTERN TABLE.
6156 033534 000000          KKPAT2: .WORD 0
6157 033536 000000          KKPAT3: .WORD 0
6158 033540 000000          KKPAT4: .WORD 0
6159 033542 000000          KKPAT5: .WORD 0
6160
6161 033544 000000          KKTMP1: .WORD 0          ;USED TO LOCATE A TEST AREA WHOSE
6162 033546 000000 000000 000000      KKTMP2: .WORD 0,0,0,0  ;HITS WON'T BE WIPE OUT BY TEST CODE.
6163 033554 000000
6164
6165 033556 010137 001636          KKERR1: MOV      R1,$TMP4  ;ERROR REPORTING ROUTINES
6166 033562 005037 001634          CLR      $TMP3
6167 033566 000427          BR       KKERR5
6168
6169 033570 010137 001636          KKERR2: MOV      R1,$TMP4
6170 033574 062737 000002 001636      ADD      #2,$TMP4
6171 033602 005037 001634          CLR      $TMP3
6172 033606 000417          BR       KKERR5
6173
6174 033610 010237 001636          KKERR3: MOV      R2,$TMP4
6175 033614 013737 000001 001634      MOV      1,$TMP3
6176 033622 000411          BR       KKERR5
6177
6178 033624 010237 001636          KKERR4: MOV      R2,$TMP4
6179 033630 062737 000002 001636      ADD      #2,$TMP4
6180 033636 012737 000001 001634      MOV      #1,$TMP3
6181 033644 000400          BR       KKERR5
6182
6183 033646 010337 001632          KKERR5: MOV      R3,$TMP2
6184 033652 011637 001630          MOV      (SP),$TMP1
6185 033656 012737 000014 177746      MOV      #MOM1,@#CONTRL
6186
6187 033664 104021          ERROR   21
6188
6189 033666 005037 177746          CLR      @#CONTRL
6190 033672 000207          RTS     PC
6191
6192 033674 104416          KK17:   RSET          ;DONE!
6193
6194          ;*****
6195          ;*TEST 34      CACHE DATA MEMORY BYTE ENABLE LOGIC TEST
6196          ;*
6197          ;*THIS TEST PERFORMS A CHECK OF THE BYTE ENABLE LOGIC
    
```


6198
 6199
 6200
 6201
 6202
 6203
 6204
 6205
 6206
 6207
 6208 033676 000004
 6209 033700 012737 000040 001676
 6210
 6211 033706 012737 035540 054230
 6212
 6213 033714 012737 054076 000114
 6214 033722 113737 001502 001626
 6215
 6216
 6217 033730 012737 001001 035376
 6218 033736 012737 004004 035400
 6219 033744 012737 020020 035402
 6220 033752 012737 100100 035404
 6221
 6222 033760 012700 035410
 6223 033764 042700 176003
 6224 033770 010001
 6225 033772 062701 140000
 6226 033776 010002
 6227 034000 062702 142000
 6228
 6229 034004 010137 001640
 6230 034010 010137 001642
 6231 034014 062737 000002 001642
 6232 034022 010237 001644
 6233 034026 010237 001646
 6234 034032 062737 000002 001646
 6235
 6236 034040 012737 034046 001512
 6237
 6238 034046 000240
 6239 034050 012737 000034 177746
 6240 034056 005711
 6241 034060 005761 000002
 6242 034064 012737 000054 177746
 6243 034072 005712
 6244 034074 005762 000002
 6245 034100 005037 177746
 6246
 6247
 6248 034104 005711
 6249
 6250
 6251 034106 032737 000010 177752
 6252 034114 001006
 6253

;*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
 MOVSB \$STSTM,\$STMP0 ;: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 #TESTR1,R1 ;:BY EXECUTING THIS TEST'S
 MOV R0,R2 ;:CODE.
 ADD #TESTR2,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
 TST (R1)
 BIT #10,@#HITMIS ;:SEE IF REFERENCE ADDRESS
 BNE MM3 ;:IS A HIT.
 ;:IF NOT ERROR.

```

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) C 12 16-MAY-79 09:11 PAGE 120
CEKBDD.P11 16-MAY-79 08:58 T34 CACHE DATA MEMORY BYTE ENABLE LOGIC TEST SEQ 0145

6254 034116 010137 001632 MOV R1,$TMP2
6255 034122 012737 000000 001630 MOV #0,$TMP1
6256 034130 104001 ERROR 1
6257
6258
6259
6260 034132 MM3:
6261
6262 034132 005761 000002 TST 2(R1)
6263
6264 ;SEE IF REFERENCE ADDRESS
6265 034136 032737 000010 177752 BIT #10,@WHITMIS ;IS A HIT.
6266 034144 001011 BNE MM4 ;IF NOT ERROR!
6267
6268 034146 010137 001632 MOV R1,$TMP2
6269 034152 062737 000002 001632 ADD #2,$TMP2
6270 034160 012737 000000 001630 MOV #0,$TMP1
6271 034166 104001 ERROR 1
6272
6273
6274
6275 034170 MM4:
6276
6277 034170 005712 TST (R2)
6278
6279 ;SEE IF REFERENCE ADDRESS
6280 034172 032737 000010 177752 BIT #10,@WHITMIS ;IS A HIT.
6281 034200 001006 BNE MM5 ;IF NOT ERROR!
6282
6283 034202 010237 001632 MOV R2,$TMP2
6284 034206 012737 000001 001630 MOV #1,$TMP1
6285 034214 104001 ERROR 1
6286
6287
6288
6289 034216 MM5:
6290
6291 034216 005762 000002 TST 2(R2)
6292
6293 ;SEE IF REFERENCE ADDRESS
6294 034222 032737 000010 177752 BIT #10,@WHITMIS ;IS A HIT.
6295 034230 001014 BNE MM6 ;IF NOT ERROR.
6296
6297 034232 010237 001632 MOV R2,$TMP2
6298 034236 062737 000002 001632 ADD #2,$TMP2
6299 034244 012737 000001 001630 MOV #1,$TMP1
6300 034252 104001 ERROR 1
6301
6302
6303
6304 034254 012737 034262 001512 MOV #MM6,$LPERR ;SET LOOP ON ERROR ADDRESS
6305 034262 012703 000001 MM6: MOV #1,R3
6306 034266 012704 000004 MM7: MOV #4,R4
6307 034272 110321 MOV R3,(R1)+ ;PUT THE TEST PATTERN
6308 034274 006103 ROL R3 ;IN GROUP 0
6309 034276 077403 SOB R4,MM7

```

6310							
6311	034300	012704	000004			MOV #4,R4	
6312	034304	110322		MM8:		MOV R3,(R2)+	;PUT THE TEST PATTERN
6313	034306	006103				ROL R3	;IN GROUP 1
6314	034310	077403				SOB R4,MM8	
6315	034312	010001				MOV R0,R1	
6316	034314	062701	140000			ADD #TESTR1,R1	;RE-ESTABLISH POINTERS TO
6317	034320	010002				MOV R0,R2	;THE TEST LOCATIONS.
6318	034322	062702	142000			ADD #TESTR2,R2	
6319	034326	012703	035376			MOV #MMPAT1,R3	;PUT THE ADDRESS OF THE TEST
6320							;PATTERN REGISTERS IN R3
6321							
6322	034332	005005				CLR R5	
6323							
6324							
6325	034334	005005				CLR R5	
6326	034336	111105				MOVB (R1),R5	;GET THE PATTERN OUT OF
6327	034340	032737	000010	177752		BIT #10,@WHITMIS	;THIS BYTE MAKING SURE IT
6328	034346	001006				BNE MM9	;IS A HIT
6329	034350	010137	001632			MOV R1,\$TMP2	
6330	034354	012737	000000	001630		MOV #0,\$TMP1	
6331	034362	104001				ERROR 1	
6332							
6333	034364	042705	177400		MM9:	BIC #177400,R5	
6334	034370	022705	000001			CMP #1,R5	;SEE IF THE DATA IS CORRECT.
6335	034374	001402				BEQ MM10	
6336	034376	004737	035420			JSR PC,MMERR1	
6337	034402	105121			MM10:	COMB (R1)+	;COMPLIMENT THE TEST PATTERN
6338	034404	012713	001376			MOV #001376,(R3)	
6339							
6340							
6341							
6342	034410	005005				CLR R5	
6343	034412	111105				MOVB (R1),R5	;GET THE PATTERN OUT OF
6344	034414	032737	000010	177752		BIT #10,@WHITMIS	;THIS BYTE MAKING SURE IT
6345	034422	001006				BNE MM11	;IS A HIT
6346	034424	010137	001632			MOV R1,\$TMP2	
6347	034430	012737	000000	001630		MOV #0,\$TMP1	
6348	034436	104001				ERROR 1	
6349							
6350	034440	042705	177400		MM11:	BIC #177400,R5	
6351	034444	022705	000002			CMP #2,R5	;SEE IF THE DATA IS CORRECT.
6352	034450	001402				BEQ MM12	
6353	034452	004737	035420			JSR PC,MMERR1	
6354	034456	105121			MM12:	COMB (R1)+	;COMPLIMENT THE TEST PATTERN
6355	034460	012713	176776			MOV #176776,(R3)	
6356							
6357							
6358	034464	062703	000002			ADD #2,R3	;POINT TO THE NEXT ELEMENT
6359							;IN THE TEST PATTERN TABLE.
6360							
6361	034470	005005				CLR R5	
6362	034472	111105				MOVB (R1),R5	;GET THE PATTERN OUT OF
6363	034474	032737	000010	177752		BIT #10,@WHITMIS	;THIS BYTE MAKING SURE IT
6364	034502	001006				BNE MM13	;IS A HIT
6365	034504	010137	001632			MOV R1,\$TMP2	

6366	034510	012737	000000	001630		MOV	#0,\$TMP1	
6367	034516	104001				ERROR	1	
6368								
6369	034520	042705	177400		MM13:	BIC	#177400,R5	
6370	034524	022705	000004			CMP	#4,R5 ;SEE IF THE DATA IS CORRECT.	
6371	034530	001402				BEQ	MM14	
6372	034532	004737	035420			JSR	PC,MMERR1	
6373	034536	105121			MM14:	COMB	(R1)+ ;COMPLIMENT THE TEST PATTERN	
6374	034540	012713	004373			MOV	#004373,(R3)	
6375								
6376								
6377								
6378	034544	005005				CLR	R5	
6379	034546	111105				MOVB	(R1),R5 ;GET THE PATTERN OUT OF	
6380	034550	032737	000010	177752		BIT	#10,@WHITMIS ;THIS BYTE MAKING SURE IT	
6381	034556	001006				BNE	MM15 ;IS A HIT	
6382	034560	010137	001632			MOV	R1,\$TMP2	
6383	034564	012737	000000	001630		MOV	#0,\$TMP1	
6384	034572	104001				ERROR	1	
6385								
6386	034574	042705	177400		MM15:	BIC	#177400,R5	
6387	034600	022705	000010			CMP	#10,R5 ;SEE IF THE DATA IS CORRECT.	
6388	034604	001402				BEQ	MM16	
6389	034606	004737	035420			JSR	PC,MMERR1	
6390	034612	105121			MM16:	COMB	(R1)+ ;COMPLIMENT THE TEST PATTERN	
6391	034614	012713	173773			MOV	#173773,(R3)	
6392								
6393								
6394	034620	062703	000002			ADD	#2,R3 ;POINT TO THE NEXT ELEMENT	
6395								;IN THE TEST PATTERN TABLE.
6396								
6397	034624	005005				CLR	R5	
6398	034626	111205				MOVB	(R2),R5 ;GET THE PATTERN OUT OF	
6399	034630	032737	000010	177752		BIT	#10,@WHITMIS ;THIS BYTE MAKING SURE IT	
6400	034636	001006				BNE	MM17 ;IS A HIT	
6401	034640	010237	001632			MOV	R2,\$TMP2	
6402	034644	012737	000001	001630		MOV	#1,\$TMP1	
6403	034652	104001				ERROR	1	
6404								
6405	034654	042705	177400		MM17:	BIC	#177400,R5	
6406	034660	022705	000020			CMP	#20,R5 ;SEE IF THE DATA IS CORRECT.	
6407	034664	001402				BEQ	MM18	
6408	034666	004737	035432			JSR	PC,MMERR2	
6409	034672	105122			MM18:	COMB	(R2)+ ;COMPLIMENT THE TEST PATTERN	
6410	034674	012713	020357			MOV	#020357,(R3)	
6411								
6412								
6413								
6414	034700	005005				CLR	R5	
6415	034702	111205				MOVB	(R2),R5 ;GET THE PATTERN OUT OF	
6416	034704	032737	000010	177752		BIT	#10,@WHITMIS ;THIS BYTE MAKING SURE IT	
6417	034712	001006				BNE	MM19 ;IS A HIT	
6418	034714	010237	001632			MOV	R2,\$TMP2	
6419	034720	012737	000001	001630		MOV	#1,\$TMP1	
6420	034726	104001				ERROR	1	
6421								

```

6422 034730 042705 177400 MM19: BIC #177400,R5
6423 034734 022705 000040 CMP #40,R5 ;SEE IF THE DATA IS CORRECT.
6424 034740 001402 BEQ MM20
6425 034742 004737 035432 JSR PC,MMERR2
6426 034746 105122 MM20: COMB (R2)+ ;COMPLIMENT THE TEST PATTERN
6427 034750 012713 157757 MOV #157757,(R3)
6428
6429
6430 034754 062703 000002 ADD #2,R3 ;POINT TO THE LAST ELEMENT
6431 ;IN THE TEST PATTERN TABLE.
6432
6433 034760 005005 CLR R5
6434 034762 111205 MOVB (R2),R5 ;GET THE PATTERN OUT OF
6435 034764 032737 000010 177752 BIT #10,@WHITMIS ;THIS BYTE MAKING SURE IT
6436 034772 001006 BNE MM21 ;IS A HIT
6437 034774 010237 001632 MOV R2,$TMP2
6438 035000 012737 000001 001630 MOV #1,$TMP1
6439 035006 104001 ERROR 1
6440
6441 035010 042705 177400 MM21: BIC #177400,R5
6442 035014 022705 000100 CMP #100,R5 ;SEE IF THE DATA IS CORRECT.
6443 035020 001402 BEQ MM22
6444 035022 004737 035432 JSR PC,MMERR2
6445 035026 105122 MM22: COMB (R2)+ ;COMPLIMENT THE TEST PATTERN
6446 035030 012713 100277 MOV #100277,(R3)
6447
6448
6449
6450 035034 005005 CLR R5
6451 035036 111205 MOVB (R2),R5 ;GET THE PATTERN OUT OF
6452 035040 032737 000010 177752 BIT #10,@WHITMIS ;THIS BYTE MAKING SURE IT
6453 035046 001006 BNE MM23 ;IS A HIT
6454 035050 010237 001632 MOV R2,$TMP2
6455 035054 012737 000001 001630 MOV #1,$TMP1
6456 035062 104001 ERROR 1
6457
6458 035064 042705 177400 MM23: BIC #177400,R5
6459 035070 022705 000200 CMP #200,R5 ;SEE IF THE DATA IS CORRECT.
6460 035074 001402 BEQ MM24
6461 035076 004737 035432 JSR PC,MMERR2
6462 035102 105122 MM24: COMB (R2)+ ;COMPLIMENT THE TEST PATTERN
6463 035104 012713 077677 MOV #077677,(R3)
6464
6465
6466 035110 010001 MOV R0,R1 ;RE-ESTABLISH POINTERS TO
6467 035112 062701 140000 ADD #TESTR1,R1 ;THE TEST AREA
6468 035116 010002 MOV R0,R2
6469 035120 062702 142000 ADD #TESTR2,R2
6470
6471
6472 035124 012105 MOV (R1)+,R5 ;CHECK THE COMPLIMENTED
6473
6474 035126 005761 177776 TST -2(R1)
6475
6476
6477 035132 032737 000010 177752 BIT #10,@WHITMIS ;SEE IF REFERENCE ADDRESS
;IS A HIT.

```

```

6478 035140 001011      BNE      MM25
6479                                ;IF NOT ERROR!
6480 035142 010137 001632      MOV      R1,$TMP2
6481 035146 062737 177776 001632      ADD      #-2,$TMP2
6482 035154 012737 000000 001630      MOV      #0,$TMP1
6483 035162 104001      ERROR    1
6484
6485
6486
6487
6488 035164 020537 035376      MM25:   CMP      R5,MMPAT1      ;IS PATTERN CORRECT?
6489 035170 001402      BEQ
6490 035172 004737 035462      JSR      PC,MMERR4
6491
6492
6493 035176      MM26:
6494
6495 035176 012105      MOV      (R1)+,R5      ;CHECK THE COMPLIMENTED
6496
6497 035200 005761 177776      TST      -2(R1)
6498
6499                                ;SEE IF REFERENCE ADDRESS
6500 035204 032737 000010 177752      BIT      #10,@#HITMIS ;IS A HIT.
6501 035212 001011      BNE      MM27
6502                                ;IF NOT ERROR
6503 035214 010137 001632      MOV      R1,$TMP2
6504 035220 062737 177776 001632      ADD      #-2,$TMP2
6505 035226 012737 000000 001630      MOV      #0,$TMP1
6506 035234 104001      ERROR    1
6507
6508
6509
6510
6511 035236 020537 035400      MM27:   CMP      R5,MMPAT2      ;IS PATTERN CORRECT?
6512 035242 001402      BEQ
6513 035244 004737 035462      JSR      PC,MMERR4
6514
6515
6516 035250      MM28:
6517
6518 035250 012205      MOV      (R2)+,R5      ;CHECK THE COMPLIMENTED
6519
6520 035252 005762 177776      TST      -2(R2)
6521
6522                                ;SEE IF REFERENCE ADDRESS
6523 035256 032737 000010 177752      BIT      #10,@#HITMIS ;IS A HIT.
6524 035264 001011      BNE      MM29
6525                                ;IF NOT ERROR.
6526 035266 010237 001632      MOV      R2,$TMP2
6527 035272 062737 177776 001632      ADD      #-2,$TMP2
6528 035300 012737 000001 001630      MOV      #1,$TMP1
6529 035306 104001      ERROR    1
6530
6531
6532
6533
    
```

```

6534 035310 020537 035402      MM29:  CMP      R5,MMPAT3      ;IS PATTERN CORRECT?
6535 035314 001402              BEQ      MM30
6536 035316 004737 035502      JSR      PC,MMERR5
6537
6538
6539 035322              MM30:
6540
6541 035322 012205              MOV      (R2)+,R5      ;CHECK THE COMPLIMENTED
6542
6543 035324 005762 177776      TST      -2(R2)
6544
6545
6546 035330 032737 000010 177752      BIT      #10,@#HITMIS ;SEE IF REFERENCE ADDRESS
6547 035336 001011              BNE      MM31          ;IS A HIT.
6548
6549 035340 010237 001632              MOV      R2,$TMP2     ;IF NOT ERROR.
6550 035344 062737 177776 001632      ADD      #-2,$TMP2
6551 035352 012737 000001 001630      MOV      #1,$TMP1
6552 035360 104001              ERROR   1
6553
6554
6555
6556
6557 035362 020537 035404      MM31:  CMP      R5,MMPAT4      ;IS PATTERN CORRECT?
6558 035366 001464              BEQ      MM32
6559 035370 004737 035502      JSR      PC,MMERR5
6560
6561 035374 000461              BR       MM32          ;FINISHED THIS TEST.
6562
6563 035376 000000      MMPAT1: .WORD 0      ;THIS IS THE TEST PATTERN
6564 035400 000000      MMPAT2: .WORD 0      ;TABLE.
6565 035402 000000      MMPAT3: .WORD 0
6566 035404 000000      MMPAT4: .WORD 0
6567
6568 035406 000000      MMTMP1: .WORD 0      ;THIS AREA IS USED TO ESTABLISH
6569 035410 000004      MMTMP2: .BLKW 4      ;A TEST LOCATION WHOSE HITS WON'T
6570
6571
6572
6573 035420 005037 001630      MMERR1: CLR      $TMP1   ;COME HERE TO REPORT
6574 035424 010137 001636      MOV      R1,$TMP4     ;GROUP 0 ERROR,WHILE READING
6575 035430 000405              BR       MMERR3       ;A BYTE INTO R5
6576
6577 035432 012737 000001 001630      MMERR2: MOV      #1,$TMP1 ;COME HERE TO REPORT
6578 035440 010237 001636      MOV      R2,$TMP4     ;GROUP 1 ERROR, READING A
6579
6580 035444 012637 001632      MMERR3: MOV      (SP)+,$TMP2 ;BYTE INTO R5.
6581 035450 010537 001634      MOV      R5,$TMP3
6582
6583 035454 104017              ERROR   17
6584 035456 000177 144150      JMP      @ $TMP2
6585
6586 035462 005037 001630      MMERR4: CLR      $TMP1   ;REPORT AN ERROR IN GROUP
6587 035466 010137 001636      MOV      R1,$TMP4     ;0 WHILE READING A WORD
6588 035472 062737 177776 001636      ADD      #-2,$TMP4
6589 035500 000410              BR       MMERR6
    
```

```

6590
6591 035502 012737 000001 001630 MMERR5: MOV #1,$TMP1
6592 035510 010237 001636 MOV R2,$TMP4
6593 035514 062737 177776 001636 ADD #-2,$TMP4
6594
6595 035522 012637 001632 MMERR6: MOV (SP)+,$TMP2
6596 035526 010537 001634 MOV R5,$TMP3
6597
6598 035532 104020 ERROR 20
6599 035534 000177 144072 JMP @TMP2
6600
6601 035540 MM32: ;DONE.

```

```

6602
6603
6604
6605
6606 *****
6607 *TEST 35 CACHE ARBITRATION AND HIGH SPEED I/O TEST
6608 *
6609 *THIS IS A TEST OF:
6610 * 1. CACHE ARBITRATION
6611 * 2. THE MASS BUS AND UNIBUS PORTS TO THE CACHE
6612 * 3. HIGH SPEED I/O THROUGH THE CACHE
6613 *
6614 *IT MAKE USE OF THE FOLLOWING DEVICES:
6615 * RS04
6616 * RP04
6617 * RK05
6618 * MASS BUSS TESTER
6619 * UNIBUS EXERCISER
6620 *
6621 *IF ANY OF THESE DEVICES ARE PRESENT AND WRITE ENABLED THE WILL BE USED
6622 *IN THIS TEST. ONLY THE LOWEST WRITE ENABLED DRIVE NUMBER OF EACH DEVICE
6623 *WILL BE USED.
6624 *
6625 * CAUTION!!
6626 * THIS TEST WILL WRITE ON THE DISKS IT USES. SO VITAL SYSTEMS
6627 * DISKS SHOULD BE REMOVED OR WRITE PROTECTED BEFORE RUNNING
6628 * THIS DIAGNOSTIC.
6629 *
6630 *IF UNIT ZERO OF A PARTICULAR DEVICE IS WRITE PROTECTED THEN THIS TEST
6631 *WILL TRY TO USE UNIT ONE, ETC.
6632 *
6633 *ALL AVAILABLE DEVICES ARE STARTED DOING TRANSFERS AT THE SAME TIME
6634 *TO DIFFERENT PARTS OF MEMORY.
6635 *EACH DEVICE HAS A CONTROL ROUTINE WHICH DRIVES THAT DEVICE THROUGH
6636 *THE CYCLE:
6637 * 1. WRITE A RANDOM DATA PATTERN IN MEMORY
6638 * 2. COPY THAT PATTERN ONTO THE DISK
6639 * 3. WRITE CHECK THE DISK
6640 * 4. READ THE PATTERN OFF THE DISK BACK INTO MEMORY
6641 * 5. CHECK DATA
6642 * 6. START OVER AT 1.
6643 *
6644 *EACH DEVICE IS CAUSED TO GO THROUGH THIS CYCLE A PREDETERMINED
6645 *NUMBER OF TIMES. THIS NUMBER IS CONTAINED IN THE LOCATION,

```



```

6646 : *CYCNT, AND CAN BE CHANGED BY THE USER AT THE CONSOLE TO ANY VALUE
6647 : *HE DESIRES).
6648 : *
6649 : *INTERRUPTS ARE ENABLED SO THAT IT IS POSSIBLE TO GET MANY DEVICES
6650 : *DOING TRANSFERS AT ONCE.
6651 : *
6652 : *UNFORTUNATELY THE DEGREE TO WHICH FAULTS CAN BE ISOLATED IS
6653 : *LIMITED BY THE FACT THAT THERE ARE MANY ELEMENTS, DEVICES, INVOLVED.
6654 : *THESE ERRORS ARE REPORTED:
6655 : *      1. ALL DEVICE ERRORS
6656 : *      2. ALL DATA OR PARITY ERRORS
6657 : *
6658 : *NOTE THAT THIS NOT INTENDED TO BE USED AS AN I/O DEVICE DIAGNOSTIC!
6659 : *ALL THE DEVICES WHICH ARE USED ARE ASSUMED TO BE IN PROPER WORKING
6660 : *CONDITION.
6661 : *
6662 : *

```

```

6663 : *****
6664 035540 000004 TST35: SCOPE
6665
6666 035542 012737 042206 054230      MOV      #TST36,SKAD      ;SET THE SKAD REGISTER
6667                                     ;IN CASE THE TEST ABORTS.
6668 035550 104416      RSET
6669 035552 113737 001502 001626      MOVB     $TSTNM,$TMP0
6670
6671 035560 012700 172340      MOV      #KIPAR0,R0      ;INITIALLY PUT MEMORY
6672 035564 012701 077406      MOV      #77406,R1      ;MANAGEMENT IN A 'PASSIVE'
6673 035570 012702 172300      MOV      #KIPDR0,R2     ;STATE, THAT IS MAP ALL
6674 035574 012703 000010      MOV      #10,R3         ;VIRTUAL ADDRESSES ON TO
6675 035600 010122 64$:      MOV      R1,(R2)+        ;THEMSELVES AS PHYSICAL
6676 035602 077302      SOB      R3,64$        ;ADDRESSES.
6677 035604 005020      CLR      (R0)+
6678 035606 012720 000200      MOV      #200,(R0)+
6679 035612 012720 000400      MOV      #400,(R0)+
6680 035616 012720 000600      MOV      #600,(R0)+
6681 035622 012720 001000      MOV      #1000,(R0)+
6682 035626 012720 001200      MOV      #1200,(R0)+
6683 035632 012720 001400      MOV      #1400,(R0)+
6684 035636 012710 177600      MOV      #177600,(R0)
6685
6686 035642 012737 000001 177572      MOV      #1,@MMR0
6687 035650 012737 000060 172516      MOV      #60,@MMR3
6688
6689 035656 004737 041752      INTO:   JSR      PC,GTBINT ;INITIALIZE THE MEMORY BUFFER
6690                                     ;ALLOCATION ROUTINES.
6691 035662 004737 055456      JSR      PC,SIZDEV      ;GO DETERMINE WHAT DEVICES ARE
6692                                     ;PRESENT.
6693 035666 005046      CLR      -(SP)         ;MAKE THE WAIT LOOP ACCESSABLE
6694 035670 012746 036140      MOV      #WAITLP,-(SP) ;TO AN 'RTI'.
6695
6696 035674 012700 056050      INT1:   MOV      #RS4DFL,R0 ;GET READY TO SEE WHAT DEVICES
6697 035700 012701 036070      MOV      #RS4CR,R1      ;ARE TO BE USED.
6698 035704 012702 036102      MOV      #RS4SUN,R2
6699 035710 012703 036114      MOV      #RS4ASS,R3
6700 035714 012704 000005      MOV      #5,R4
6701

```

```

6702 035720 005011          INT2: CLR (R1)          ;CLEAR THE UNIT NUMBER.
6703 035722 005012          CLR (R2)          ;CLEAR THE COUNTER.
6704 035724 105710          TSTB (R0)         ;IS THERE A DRIVE.
6705 035726 001447          BEQ INT6         ;BRANCH IF NOT.
6706
6707 035730 111005          MOVB (R0),R5      ;OTHERWISE DETERMINE A UNIT NUM.
6708 035732 104412          SAVREG
6709 035734 012700 000010   MOV #10,R0
6710 035740 005001          CLR R1
6711 035742 012702 000001   MOV #1,R2
6712 035746 030205          INT3: BIT R2,R5
6713 035750 001405          BEQ INT4
6714 035752 010137 036064   MOV R1,INTMP1
6715 035756 104414          RESREG
6716 035760 000137 036000   JMP INT5
6717 035764 005201          INT4: INC R1
6718 035766 006302          ASL R2
6719 035770 077012          SOB R0,INT3
6720 035772 104414          RESREG
6721 035774 000137 036046   JMP INT6
6722
6723 036000 013711 036066   INT5: MOV CYCNT,(R1)      ;FOUND THE DRIVE SO SET UP THE
6724 036004 020127 036070   CMP R1,#RS4CR
6725 036010 001001          BNE 1$
6726 036012 006311          ASL (R1)
6727 036014 020127 036072   1$: CMP R1,#RP4CR
6728 036020 001001          BNE 2$
6729 036022 006311          ASL (R1)
6730 036024 020127 036074   2$: CMP R1,#RH4CR
6731 036030 001001          BNE 3$
6732 036032 006311          ASL (R1)
6733 036034 012746 000340   3$: MOV #340,-(SP)      ;PASS COUNT AND MAKE THE DRIVER
6734 036040 011346          MOV (R3),-(SP)   ;ACCESSIBLE BY A 'RTI'.
6735 036042 013712 036064   MOV INTMP1,(R2)
6736
6737 036046 005200          INT6: INC R0
6738 036050 005721          TST (R1)+
6739 036052 022223          CMP (R2)+,(R3)+ ;MOVE THE POINTERS TO THE NEXT DEVICE.
6740 036054 000240          NOP
6741 036056 077460          SOB R4,INT2
6742
6743
6744 036060 000240          NOP
6745 036062 000002          RTI              ;START THE TEST.
6746
6747
6748 ;THESE ARE SOME TABLES THAT ARE USED TO CONTROL AND SET UP THIS TEST.
6749 036064 000000          INTMP1: .WORD 0
6750
6751
6752 036066 000010          CYCNT: .WORD 10 ;THE PASS COUNT.!!!
6753
6754 036070 000000          RS4CR: .WORD 0 ;PASS COUNT FOR EACH DEVICE.
6755 036072 000000          RP4CR: .WORD 0
6756 036074 000000          RH4CR: .WORD 0
6757 036076 000000          RK5CR: .WORD 0
    
```

```

6758 036100 000000      UBECR: .WORD 0
6759
6760 036102 000000      RS4SUN: .WORD 0      ;THE DRIVE NUMBER USED FOR EACH
6761 036104 000000      RP4SUN: .WORD 0      ;DEVICE.
6762 036106 000000      RH4SUN: .WORD 0
6763 036110 000000      RK5SUN: .WORD 0
6764 036112 000000      UBESUN: .WORD 0
6765
6766          036114      SETBLE=RS4ASS
6767 036114 036204      RS4ASS: .WORD DRRS4      ;STARTING ADDRESSES OF EACH DRIVER.
6768 036116 037016      RP4ASS: .WORD DRRP4
6769 036120 037630      RH4ASS: .WORD DRRH4
6770 036122 040422      RK5ASS: .WORD DRRK5
6771 036124 041234      UBEASS: .WORD DRUBE
6772
6773 036126 000000      RS4RB: .WORD 0      ;WRITE AND READ BUFFERS OF EACH DEVICE.
6774 036130 000000      RP4RB: .WORD 0
6775 036132 000000      RH4RB: .WORD 0
6776 036134 000000      RK5RB: .WORD 0
6777 036136 000000      UBERB: .WORD 0
6778
6779
6780      ;THIS IS THE WAIT ROUTINE. COME HERE WHEN WAITING FOR AN INTERRUPT
6781      ;OR WHEN DONE, ALL THE PASS COUNTS HAVE GONE TO ZERO.
6782 036140 000230      WAITLP: SPL 0      ;LOWER THE PRIORITY.
6783 036142 005737 036076      TST RK5CR      ;WAIT FOR INTERRUPT OR ZERO PASS COUNT.
6784 036146 001374      BNE WAITLP
6785 036150 005737 036100      TST UBECR
6786 036154 001371      BNE WAITLP
6787 036156 005737 036072      TST RP4CR
6788 036162 001366      BNE WAITLP
6789 036164 005737 036070      TST RS4CR
6790 036170 001363      BNE WAITLP
6791 036172 005737 036074      TST RH4CR
6792 036176 001360      BNE WAITLP
6793
6794 036200 000137 042204      JMP INDONE      ;FINISHED...
6795
6796
6797
6798      ;THIS IS THE RS4 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
6799      ;TEST.
6800
6801 036204 000240      DRRS4: NOP
6802 036206 012737 007007 037012      MOV #7007,DRS4T1      ;INITIALIZE THE RANDOM DISK ADDRESS
6803 036214 012737 006006 037014      MOV #6006,DRS4T2      ;GENERATER.
6804 036222 012737 005005 036466      MOV #5005,RS4AA3
6805
6806 036230 000240      RS4AA: NOP
6807 036232 000240      NOP
6808 036234 104412      SAVREG
6809 036236 004737 042066      JSR PC,GETBUF      ;GET A MEMORY BUFFER.
6810 036242 036126      .WORD RS4RB
6811 036244 013701 036126      MOV RS4RB,R1
6812 036250 005000      CLR R0
6813 036252 073027 000014      ASHC #12.,R0

```

```

6814
6815 036256 000237          SPL      7          ;GET A RANDOM DISK ADDRESS.
6816 036260 013737 037012 053106  MOV     DRS4T1,$HINUM
6817 036266 013737 037014 053110  MOV     DRS4T2,$LONUM
6818 036274 004737 053010          JSR     PC,$RAND
6819 036300 013737 053106 037012  MOV     $HINUM,DRS4T1
6820 036306 013737 053110 037014  MOV     $LONUM,DRS4T2
6821 036314 000230          SPL      0
6822
6823 036316 013702 036102          MOV     RS4SUN,R2          ;SET UP THE DEVICE UNIT NUM.
6824 036322 110237 036663  MOVVB  R2,RS4112
6825 036326 110237 036511  MOVVB  R2,RS4BB
6826 036332 110237 036555  MOVVB  R2,RS4HH
6827 036336 110237 036621  MOVVB  R2,RS4NN
6828
6829 036342 013703 037012          MOV     DRS4T1,R3          ;SET UP THE DISK ADDRESS.
6830 036346 013704 037014  MOV     DRS4T2,R4
6831 036352 010337 036512  MOV     R3,RS4CC
6832 036356 010337 036664  MOV     R3,RS4113
6833 036362 010337 036556  MOV     R3,RS4II
6834 036366 010337 036622  MOV     R3,RS400
6835 036372 010437 036514  MOV     R4,RS4DD
6836 036376 010437 036560  MOV     R4,RS4JJ
6837 036402 010437 036666  MOV     R4,RS4114
6838 036406 010437 036624  MOV     R4,RS4PP
6839
6840 036412 010137 036470          MOV     R1,RS4AA1          ;SET THE MEMORY ADDRESS.
6841 036416 010137 036516  MOV     R1,RS4EE
6842 036422 010137 036562  MOV     R1,RS4KK
6843 036426 010137 036626  MOV     R1,RS4QQ
6844 036432 010137 036670  MOV     R1,RS4115
6845 036436 010037 036630  MOV     R0,RS4RR
6846 036442 010037 036672  MOV     R0,RS4116
6847 036446 010037 036472  MOV     R0,RS4AA2
6848 036452 010037 036520  MOV     R0,RS4FF
6849 036456 010037 036564  MOV     R0,RS4LL
6850
6851 036462 104414          RESREG
6852
6853 036464 104440          WRRAND          ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
6854 036466 000000  RS4AA3: .WORD 0
6855 036470 000000  RS4AA1: .WORD 0
6856 036472 000000  RS4AA2: .WORD 0
6857 036474 004000          .WORD 4000
6858 036476 005237 036466  INC     RS4AA3
6859
6860 036502 000240          NOP
6861 036504 000237          SPL      7
6862 036506 104442          CALRS4          ;GET THE RS4 TO DO THE TRANSFER FROM MEMORY
6863 036510          161          .BYTE 161
6864 036511          000          RS4BB: .BYTE 0
6865 036512 000000          RS4CC: .WORD 0
6866 036514 000000          RS4DD: .WORD 0
6867 036516 000000          RS4EE: .WORD 0
6868 036520 000000          RS4FF: .WORD 0
6869 036522 004000          .WORD 4000
    
```

```

6870 036524 036542 .WORD RS4GG
6871
6872 036526 000240 NOP
6873 036530 004737 036740 JSR PC,RS4YY
6874 036534 005066 000002 CLR 2(SP)
6875 036540 000002 RTI ;GO DO SOMETHING ELSE WHILE WAITING
6876 ;FOR THE INTERRUPT!
6877
6878 036542 000240 RS4GG: NOP
6879 036544 004737 036740 JSR PC,RS4YY ;SEE IF THERE WERE ANY ERRORS.
6880
6881 036550 000237 SPL 7
6882 036552 104442 CALRS4 ;DO THE WRITE CHECK
6883 036554 151 .BYTE 151
6884 036555 000 RS4HH: .BYTE 0
6885 036556 000000 RS4II: .WORD 0
6886 036560 000000 RS4JJ: .WORD 0
6887 036562 000000 RS4KK: .WORD 0
6888 036564 000000 RS4LL: .WORD 0
6889 036566 004000 .WORD 4000
6890 036570 036606 .WORD RS4MM
6891
6892 036572 000240 NOP
6893 036574 004737 036740 JSR PC,RS4YY
6894 036600 005066 000002 CLR 2(SP)
6895 036604 000002 RTI ;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
6896
6897 036606 000240 RS4MM: NOP
6898 036610 004737 036740 JSR PC,RS4YY ;SEE IF THERE WERE ANY ERRORS.
6899
6900
6901 036614 000237 SPL 7
6902 036616 104442 CALRS4 ;READ THE DISK.
6903 036620 171 .BYTE 171
6904 036621 000 RS4NN: .BYTE 0
6905 036622 000000 RS4OO: .WORD 0
6906 036624 000000 RS4PP: .WORD 0
6907 036626 000000 RS4QQ: .WORD 0
6908 036630 000000 RS4RR: .WORD 0
6909 036632 004000 .WORD 4000
6910 036634 036652 .WORD RS4111
6911
6912 036636 000240 NOP
6913 036640 004737 036740 JSR PC,RS4YY
6914 036644 005066 000002 CLR 2(SP)
6915 036650 000002 RTI ;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
6916
6917 036652 004737 036740 RS4111: JSR PC,RS4YY
6918 036656 000237 SPL 7
6919
6920 036660 104442 CALRS4
6921 036662 151 .BYTE 151
6922 036663 000 RS4112: .BYTE 0
6923 036664 000000 RS4113: .WORD 0
6924 036666 000000 RS4114: .WORD 0
6925 036670 000000 RS4115: .WORD 0
    
```

```

6926 036672 000000 RS4116: .WORD 0
6927 036674 004000 .WORD 4000
6928 036676 036714 .WORD RS4SS
6929 036700 000240 NOP
6930 036702 004737 036740 JSR PC,RS4YY
6931 036706 005066 000002 CLR 2(SP)
6932 036712 000002 RTI
6933
6934 036714 000240 RS4SS: NOP
6935 036716 004737 036740 JSR PC,RS4YY ;SEE IF ANY ERRORS OCCURRED.
6936
6937 036722 005337 036070 DEC RS4CR ;DECRIEMENT THE PASS COUNT.
6938 036726 001001 BNE RS4XX ;IF NOT DONE CONTINUE.
6939 036730 000002 RTI ;IF DONE GET OUT.
6940
6941 036732 000240 RS4XX: NOP
6942 036734 000137 036230 JMP RS4AA ;RESTART.
6943
6944 036740 000240 RS4YY: NOP
6945 036742 005737 057374 TST RS4ER1 ;SEE IF ANY ERRORS OCCURRED.
6946 036746 001420 BEQ RS4ZZ ;IF NOT THEN RETURN TO CALL.
6947
6948 036750 000237 SPL 7
6949 036752 005037 036070 CLR RS4CR ;IF YES THEN CLEAR THE PASS COUNT.
6950 036756 013737 057376 001630 MOV RS4ER2,$TMP1 ;AND MAKE AN ERROR CALL.
6951 036764 013737 057402 001634 MOV RS4ER4,$TMP3
6952 036772 013737 057400 001632 MOV RS4ER3,$TMP2
6953 037000 104154 ERROR 154
6954 037002 000230 SPL 0
6955 037004 005726 TST (SP)+
6956 037006 000002 RTI ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
6957 ;FROM THE TEST.
6958
6959 037010 000207 RS4ZZ: RTS PC ;THERE WERE NO ERRORS.
6960
6961 037012 000000 DRS4T1: .WORD 0
6962 037014 000000 DRS4T2: .WORD 0
6963
6964
6965
6966 ;THIS IS THE RP4 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
6967 ;TEST.
6968
6969 037016 000240 DRRP4: NOP
6970 037020 012737 004004 037624 MOV #4004,DRP4T1 ;INITIALIZE THE RANDOM DISK ADDRESS
6971 037026 012737 003003 037626 MOV #3003,DRP4T2 ;GENERATER.
6972 037034 012737 002002 037300 MOV #2002,RP4AA3
6973
6974 037042 000240 RP4AA: NOP
6975 037044 000240 NOP
6976 037046 104412 SAVREG
6977 037050 004737 042066 JSR PC,GETBUF ;GET A MEMORY BUFFER.
6978 037054 036130 .WORD RP4RB
6979 037056 013701 036130 MOV RP4RB,R1
6980 037062 005000 CLR R0
6981 037064 073027 000014 ASHC #12.,R0
    
```

```

6982
6983 037070 000237          SPL      7          ;GET A RANDOM DISK ADDRESS.
6984 037072 013737 037624 053106  MOV     DRP4T1,$HINUM
6985 037100 013737 037626 053110  MOV     DRP4T2,$LONUM
6986 037106 004737 053010          JSR     PC,$RAND
6987 037112 013737 053106 037624  MOV     $HINUM,DRP4T1
6988 037120 013737 053110 037626  MOV     $LONUM,DRP4T2
6989 037126 000230          SPL      0
6990
6991 037130 013702 036104          MOV     RP4SUN,R2          ;SET UP THE DEVICE UNIT NUM.
6992 037134 110237 037475  MOVVB  R2,RP4I12
6993 037140 110237 037323  MOVVB  R2,RP4BB
6994 037144 110237 037367  MOVVB  R2,RP4HH
6995 037150 110237 037433  MOVVB  R2,RP4NN
6996
6997 037154 013703 037624          MOV     DRP4T1,R3          ;SET UP THE DISK ADDRESS.
6998 037160 013704 037626  MOV     DRP4T2,R4
6999 037164 010337 037324  MOV     R3,RP4CC
7000 037170 010337 037476  MOV     R3,RP4I13
7001 037174 010337 037370  MOV     R3,RP4I1
7002 037200 010337 037434  MOV     R3,RP400
7003 037204 010437 037326  MOV     R4,RP4DD
7004 037210 010437 037372  MOV     R4,RP4JJ
7005 037214 010437 037500  MOV     R4,RP4I14
7006 037220 010437 037436  MOV     R4,RP4PP
7007
7008 037224 010137 037302          MOV     R1,RP4AA1          ;SET THE MEMORY ADDRESS.
7009 037230 010137 037330  MOV     R1,RP4EE
7010 037234 010137 037374  MOV     R1,RP4KK
7011 037240 010137 037440  MOV     R1,RP400
7012 037244 010137 037502  MOV     R1,RP4I15
7013 037250 010037 037442  MOV     R0,RP4RR
7014 037254 010037 037504  MOV     R0,RP4I16
7015 037260 010037 037304  MOV     R0,RP4AA2
7016 037264 010037 037332  MOV     R0,RP4FF
7017 037270 010037 037376  MOV     R0,RP4LL
7018
7019 037274 104414          RESREG
7020
7021 037276 104440          WRRAND          ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
7022 037300 000000  RP4AA3: .WORD 0
7023 037302 000000  RP4AA1: .WORD 0
7024 037304 000000  RP4AA2: .WORD 0
7025 037306 004000          .WORD 4000
7026 037310 005237 037300  INC     RP4AA3
7027
7028 037314 000240          NOP
7029 037316 000237          SPL      7
7030 037320 104444          CALRP4          ;GET THE RP4 TO DO THE TRANSFER FROM MEMORY
7031 037322 161          .BYTE 161
7032 037323 000          RP4BB: .BYTE 0
7033 037324 000000  RP4CC: .WORD 0
7034 037326 000000  RP4DD: .WORD 0
7035 037330 000000  RP4EE: .WORD 0
7036 037332 000000  RP4FF: .WORD 0
7037 037334 004000          .WORD 4000
    
```

```

7038 037336 037354 .WORD RP4GG
7039
7040 037340 000240 NOP
7041 037342 004737 037552 JSR PC,RP4YY
7042 037346 005066 000002 CLR 2(SP)
7043 037352 000002 RTI ;GO DO SOMETHING ELSE WHILE WAITING
7044 ;FOR THE INTERRUPT!
7045
7046 037354 000240 RP4GG: NOP
7047 037356 004737 037552 JSR PC,RP4YY ;SEE IF THERE WERE ANY ERRORS.
7048
7049 037362 000237 SPL 7
7050 037364 104444 CALRP4 ;DO THE WRITE CHECK
7051 037366 151 .BYTE 151
7052 037367 000 RP4HH: .BYTE 0
7053 037370 000000 RP4II: .WORD 0
7054 037372 000000 RP4JJ: .WORD 0
7055 037374 000000 RP4KK: .WORD 0
7056 037376 000000 RP4LL: .WORD 0
7057 037400 004000 .WORD 4000
7058 037402 037420 .WORD RP4MM
7059
7060 037404 000240 NOP
7061 037406 004737 037552 JSR PC,RP4YY
7062 037412 005066 000002 CLR 2(SP)
7063 037416 000002 RTI ;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
7064
7065 037420 000240 RP4MM: NOP
7066 037422 004737 037552 JSR PC,RP4YY ;SEE IF THERE WERE ANY ERRORS.
7067
7068
7069 037426 000237 SPL 7
7070 037430 104444 CALRP4 ;READ THE DISK.
7071 037432 171 .BYTE 171
7072 037433 000 RP4NN: .BYTE 0
7073 037434 000000 RP4OO: .WORD 0
7074 037436 000000 RP4PP: .WORD 0
7075 037440 000000 RP4QQ: .WORD 0
7076 037442 000000 RP4RR: .WORD 0
7077 037444 004000 .WORD 4000
7078 037446 037464 .WORD RP4111
7079
7080 037450 000240 NOP
7081 037452 004737 037552 JSR PC,RP4YY
7082 037456 005066 000002 CLR 2(SP)
7083 037462 000002 RTI ;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
7084
7085 037464 004737 037552 RP4111: JSR PC,RP4YY
7086 037470 000237 SPL 7
7087
7088 037472 104444 CALRP4
7089 037474 151 .BYTE 151
7090 037475 000 RP4112: .BYTE 0
7091 037476 000000 RP4113: .WORD 0
7092 037500 000000 RP4114: .WORD 0
7093 037502 000000 RP4115: .WORD 0
    
```



```

7094 037504 000000      RP4116: .WORD 0
7095 037506 004000      .WORD 4000
7096 037510 037526      .WORD RP4SS
7097 037512 000240      NOP
7098 037514 004737 037552 JSR PC,RP4YY
7099 037520 005066 000002 CLR 2(SP)
7100 037524 000002      RTI
7101
7102 037526 000240      RP4SS: NOP
7103 037530 004737 037552 JSR PC,RP4YY ;SEE IF ANY ERRORS OCCURRED.
7104
7105 037534 005337 036072 DEC RP4CR ;DECRIEMENT THE PASS COUNT.
7106 037540 001001      BNE RP4XX ;IF NOT DONE CONTINUE.
7107 037542 000002      RTI ;IF DONE GET OUT.
7108
7109 037544 000240      RP4XX: NOP
7110 037546 000137 037042 JMP RP4AA ;RESTART.
7111
7112 037552 000240      RP4YY: NOP
7113 037554 005737 056424 TST RP4ER1 ;SEE IF ANY ERRORS OCCURRED.
7114 037560 001420      BEQ RP4ZZ ;IF NOT THEN RETURN TO CALL.
7115
7116 037562 000237      SPL 7
7117 037564 005037 036072 CLR RP4CR ;IF YES THEN CLEAR THE PASS COUNT.
7118 037570 013737 056426 001630 MOV RP4ER2,$TMP1 ;AND MAKE AN ERROR CALL.
7119 037576 013737 056432 001634 MOV RP4ER4,$TMP3
7120 037604 013737 056430 001632 MOV RP4ER3,$TMP2
7121 037612 104155      ERROR 155
7122 037614 000230      SPL 0
7123 037616 005726      TST (SP)+
7124 037620 000002      RTI ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
7125 ;FROM THE TEST.
7126
7127 037622 000207      RP4ZZ: RTS PC ;THERE WERE NO ERRORS.
7128
7129 037624 000000      DRP4T1: .WORD 0
7130 037626 000000      DRP4T2: .WORD 0
7131
7132
7133
7134 ;THIS IS THE RH4 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
7135 ;TEST.
7136
7137 037630 000240      DRRH4: NOP
7138 037632 012737 070070 040416 MCV #70070,DRH4T1 ;INITIALIZE THE RANDOM DISK ADDRESS
7139 037640 012737 060060 040420 MOV #60060,DRH4T2 ;GENERATER.
7140 037646 012737 050050 040072 MCV #50050,RH4AA3
7141
7142 037654 000240      RH4AA: NOP
7143 037656 000240      NOP
7144 037660 104412      SAVREG
7145 037662 004737 042066 JSR PC,GETBUF ;GET A MEMORY BUFFER.
7146 037666 036132      .WORD RH4RB
7147 037670 013701 036132 MOV RH4RB,R1
7148 037674 005000      CLR R0
7149 037676 073027 000014 ASHC #12.,R0

```

```

7150
7151 037702 000237          SPL      7          ;GET A RANDOM DISK ADDRESS.
7152 037704 013737 040416 053106      MOV      DRH4T1,$HINUM
7153 037712 013737 040420 053110      MOV      DRH4T2,$LONUM
7154 037720 004737 053010          JSR      PC,$RAND
7155 037724 013737 053106 040416      MOV      $HINUM,DRH4T1
7156 037732 013737 053110 040420      MOV      $LONUM,DRH4T2
7157 037740 000230          SPL      0
7158
7159 037742 013702 036106          MOV      RH4SUN,R2          ;SET UP THE DEVICE UNIT NUM.
7160 037746 110237 040267          MOVB    R2,RH4112
7161 037752 110237 040115          MOVB    R2,RH4BB
7162 037756 110237 040161          MOVB    R2,RH4HH
7163 037762 110237 040225          MOVB    R2,RH4NN
7164
7165 037766 013703 040416          MOV      DRH4T1,R3          ;SET UP THE DISK ADDRESS.
7166 037772 013704 040420          MOV      DRH4T2,R4
7167 037776 010337 040116          MOV      R3,RH4CC
7168 040002 010337 040270          MOV      R3,RH4113
7169 040006 010337 040162          MOV      R3,RH4II
7170 040012 010337 040226          MOV      R3,RH400
7171
7172 040016 010137 040074          MOV      R1,RH4AA1          ;SET THE MEMORY ADDRESS.
7173 040022 010137 040122          MOV      R1,RH4EE
7174 040026 010137 040166          MOV      R1,RH4KK
7175 040032 010137 040232          MOV      R1,RH4QQ
7176 040036 010137 040274          MOV      R1,RH4115
7177 040042 010037 040234          MOV      R0,RH4RR
7178 040046 010037 040276          MOV      R0,RH4116
7179 040052 010037 040076          MOV      R0,RH4AA2
7180 040056 010037 040124          MOV      R0,RH4FF
7181 040062 010037 040170          MOV      R0,RH4LL
7182
7183 040066 104414          RESREG
7184
7185 040070 104440          WRRAND          ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
7186 040072 000000          RH4AA3: .WORD 0
7187 040074 000000          RH4AA1: .WORD 0
7188 040076 000000          RH4AA2: .WORD 0
7189 040100 004000          .WORD 4000
7190 040102 005237 040072          INC     RH4AA3
7191
7192 040106 000240          NOP
7193 040110 000237          SPL      7
7194 040112 104446          CALRH4          ;GET THE RH4 TO DO THE TRANSFER FROM MEMORY
7195 040114 161          .BYTE 16'
7196 040115 000          RH4BB: .BYTE 0
7197 040116 000000          RH4CC: .WORD 0
7198 040120 000000          RH4DD: .WORD 0
7199 040122 000000          RH4EE: .WORD 0
7200 040124 000000          RH4FF: .WORD 0
7201 040126 004000          .WORD 4000
7202 040130 040146          .WORD RH4GG
7203
7204 040132 000240          NOP
7205 040134 004737 040344          JSR      PC,RH4YY

```

```

7206 040140 005066 000002 CLR 2(SP)
7207 040144 000002 RTI ;GO DO SOMETHING ELSE WHILE WAITING
7208 ;FOR THE INTERRUPT.
7209
7210 040146 000240 RH4GG: NOP
7211 040150 004737 040344 JSR PC,RH4YY ;SEE IF THERE WERE ANY ERRORS.
7212
7213 040154 000237 SPL 7
7214 040156 104446 CALRH4 ;DO THE WRITE CHECK
7215 040160 171 .BYTE 171
7216 040161 000 RH4HH: .BYTE 0
7217 040162 000000 RH4II: .WORD 0
7218 040164 000000 RH4JJ: .WORD 0
7219 040166 000000 RH4KK: .WORD 0
7220 040170 000000 RH4LL: .WORD 0
7221 040172 004000 .WORD 4000
7222 040174 040212 .WORD RH4MM
7223
7224 040176 000240 NOP
7225 040200 004737 040344 JSR PC,RH4YY
7226 040204 005066 000002 CLR 2(SP)
7227 040210 000002 RTI ;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
7228
7229 040212 000240 RH4MM: NOP
7230 040214 004737 040344 JSR PC,RH4YY ;SEE IF THERE WERE ANY ERRORS.
7231
7232
7233 040220 000237 SPL 7
7234 040222 104446 CALRH4 ;READ THE DISK.
7235 040224 151 .BYTE 151
7236 040225 000 RH4NN: .BYTE 0
7237 040226 000000 RH4OO: .WORD 0
7238 040230 000000 RH4PP: .WORD 0
7239 040232 000000 RH4QQ: .WORD 0
7240 040234 000000 RH4RR: .WORD 0
7241 040236 004000 .WORD 4000
7242 040240 040256 .WORD RH4111
7243
7244 040242 000240 NOP
7245 040244 004737 040344 JSR PC,RH4YY
7246 040250 005066 000002 CLR 2(SP)
7247 040254 000002 RTI ;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
7248
7249 040256 004737 040344 RH4111: JSR PC,RH4YY
7250 040262 000237 SPL 7
7251
7252 040264 104446 CALRH4
7253 040266 171 .BYTE 171
7254 040267 000 RH4112: .BYTE 0
7255 040270 000000 RH4113: .WORD 0
7256 040272 000000 RH4114: .WORD 0
7257 040274 000000 RH4115: .WORD 0
7258 040276 000000 RH4116: .WORD 0
7259 040300 004000 .WORD 4000
7260 040302 040320 .WORD RH4SS
7261 040304 000240 NOP

```

```

7262 040306 004737 040344      JSR    PC,RH4YY
7263 040312 005066 000002      CLR    2(SP)
7264 040316 000002
7265
7266 040320 000240      RH455: NOP
7267 040322 004737 040344      JSR    PC,RH4YY      ;SEE IF ANY ERRORS OCCURRED.
7268
7269 040326 005337 036074      DEC    RH4CR      ;DECRIEMENT THE PASS COUNT.
7270 040332 001001      BNE    RH4XX      ;IF NOT DONE CONTINUE.
7271 040334 000002      RTI    ;IF DONE GET OUT.
7272
7273 040336 000240      RH4XX: NOP
7274 040340 000137 037654      JMP    RH4AA      ;RESTART.
7275
7276 040344 000240      RH4YY: NOP
7277 040346 005737 062076      TST    RH4ER1 ;SEE IF ANY ERRORS OCCURRED.
7278 040352 001420      BEQ    RH4ZZ      ;IF NOT THEN RETURN TO CALL.
7279
7280 040354 000237      SPL    7
7281 040356 005037 036074      CLR    RH4CR      ;IF YES THEN CLEAR THE PASS COUNT.
7282 040362 013737 062100 001630      MOV    RH4ER2,$TMP1 ;AND MAKE AN ERROR CALL.
7283 040370 013737 062104 001634      MOV    RH4ER4,$TMP3
7284 040376 013737 062102 001632      MCV    RH4ER3,$TMP2
7285 040404 104156      ERROR  156
7286 040406 000230      SPL    0
7287 040410 005726      TST    (SP)+
7288 040412 000002      RTI    ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
7289
7290
7291 040414 000207      RH4ZZ: RTS    PC      ;THERE WERE NO ERRORS.
7292
7293 040416 000000      DRH4T1: .WORD 0
7294 040420 000000      DRH4T2: .WORD 0
7295
7296
7297
7298
7299
7300      ;THIS IS THE RK5 DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
7301      ;TEST.
7302 040422 000240      DRRK5: NOP
7303 040424 012737 030030 041230      MOV    #30030,DRK5T1 ;INITIALIZE THE RANDOM DISK ADDRESS
7304 040432 012737 040040 041232      MOV    #40040,DRK5T2 ;GENERATER.
7305 040440 012737 050050 040704      MOV    #50050,RK5AA3
7306
7307 040446 000240      RK5AA: NOP
7308 040450 000240      NOP
7309 040452 104412      SAVREG
7310 040454 004737 042066      JSR    PC,GETBUF    ;GET A MEMORY BUFFER.
7311 040460 036134      .WORD  RK5RB
7312 040462 013701 036134      MOV    RK5RB,R1
7313 040466 005000      CLR    R0
7314 040470 073027 000014      ASHC   #12.,R0
7315
7316 040474 000237      SPL    7      ;GET A RANDOM DISK ADDRESS.
7317 040476 013737 041230 053106      MOV    DRK5T1,$HINUM

```

```

7318 040504 013737 041232 053110      MOV      DRK5T2,$LONUM
7319 040512 004737 053010      JSR      PC,$RAND
7320 040516 013737 053106 041230      MOV      $HINUM,DRK5T1
7321 040524 013737 053110 041232      MOV      $LONUM,DRK5T2
7322 040532 000230      SPL      0
7323
7324 040534 013702 036110      MOV      RK5SUN,R2          ;SET UP THE DEVICE UNIT NUM.
7325 040540 110237 041101      MOV      R2,RK5112
7326 040544 110237 040727      MOV      R2,RK5BB
7327 040550 110237 040773      MOV      R2,RK5HH
7328 040554 110237 041037      MOV      R2,RK5NN
7329
7330 040560 013703 041230      MOV      DRK5T1,R3          ;SET UP THE DISK ADDRESS.
7331 040564 013704 041232      MOV      DRK5T2,R4
7332 040570 010337 040730      MOV      R3,RK5CC
7333 040574 010337 041102      MOV      R3,RK5113
7334 040600 010337 040774      MOV      R3,RK511
7335 040604 010337 041040      MOV      R3,RK500
7336 040610 010437 040732      MOV      R4,RK5DD
7337 040614 010437 040776      MOV      R4,RK5JJ
7338 040620 010437 041104      MOV      R4,RK5114
7339 040624 010437 041042      MOV      R4,RK5PP
7340
7341 040630 010137 040706      MOV      R1,RK5AA1          ;SET THE MEMORY ADDRESS.
7342 040634 010137 040734      MOV      R1,RK5EE
7343 040640 010137 041000      MOV      R1,RK5KK
7344 040644 010137 041044      MOV      R1,RK5QQ
7345 040650 010137 041106      MOV      R1,RK5115
7346 040654 010037 041046      MOV      R0,RK5RR
7347 040660 010037 041110      MOV      R0,RK5116
7348 040664 010037 040710      MOV      R0,RK5AA2
7349 040670 010037 040736      MOV      R0,RK5FF
7350 040674 010037 041002      MOV      R0,RK5LL
7351
7352 040700 104414      RESREG
7353
7354 040702 104440      WRRAND          ;FILL THE MEMORY BUFFER WITH RANDOM DATA.
7355 040704 000000      RK5AA3: .WORD 0
7356 040706 000000      RK5AA1: .WORD 0
7357 040710 000000      RK5AA2: .WORD 0
7358 040712 004000      .WORD 4000
7359 040714 005237 040704      INC      RK5AA3
7360
7361 040720 000240      NOP
7362 040722 000237      SPL      7
7363 040724 104450      CALRKS          ;GET THE RK5 TO DO THE TRANSFER FROM MEMORY
7364 040726 103      .BYT 103
7365 040727 000      RK5BB: .BYTE 0
7366 040730 000000      RK5CC: .WORD 0
7367 040732 000000      RK5DD: .WORD 0
7368 040734 000000      RK5EE: .WORD 0
7369 040736 000000      RK5FF: .WORD 0
7370 040740 004000      .WORD 4000
7371 040742 040760      .WORD RK5GG
7372
7373 040744 000240      NOP
    
```

7374	040746	004737	041156		JSR	PC,RK5YY	
7375	040752	005066	000002		CLR	2(SP)	
7376	040756	000002			RTI		;GO DO SOMETHING ELSE WHILE WAITING ;FOR THE INTERRUPT!
7377							
7378							
7379	040760	000240		RK5GG:	NOP		
7380	040762	004737	041156		JSR	PC,RK5YY	;SEE IF THERE WERE ANY ERRORS.
7381							
7382	040766	000237			SPL	7	
7383	040770	104450			CALRK5		;DO THE WRITE CHECK
7384	040772	107			.BYTE	107	
7385	040773	000		RK5HH:	.BYTE	0	
7386	040774	000000		RK5II:	.WORD	0	
7387	040776	000000		RK5JJ:	.WORD	0	
7388	041000	000000		RK5KK:	.WORD	0	
7389	041002	000000		RK5LL:	.WORD	0	
7390	041004	004000			.WORD	4000	
7391	041006	041024			.WORD	RK5MM	
7392							
7393	041010	000240			NOP		
7394	041012	004737	041156		JSR	PC,RK5YY	
7395	041016	005066	000002		CLR	2(SP)	
7396	041022	000002			RTI		;DO SOMETHING ELSE WHILE WAITING FOR INTERRUPT.
7397							
7398	041024	000240		RK5MM:	NOP		
7399	041026	004737	041156		JSR	PC,RK5YY	;SEE IF THERE WERE ANY ERRORS.
7400							
7401							
7402	041032	000237			SPL	7	
7403	041034	104450			CALRK5		;READ THE DISK.
7404	041036	105			.BYTE	105	
7405	041037	000		RK5NN:	.BYTE	0	
7406	041040	000000		RK5OO:	.WORD	0	
7407	041042	000000		RK5PP:	.WORD	0	
7408	041044	000000		RK5QQ:	.WORD	0	
7409	041046	000000		RK5RR:	.WORD	0	
7410	041050	004000			.WORD	4000	
7411	041052	041070			.WORD	RK5111	
7412							
7413	041054	000240			NOP		
7414	041056	004737	041156		JSR	PC,RK5YY	
7415	041062	005066	000002		CLR	2(SP)	
7416	041066	000002			RTI		;DO SOMETHING ELSE WHILE WAITING FOR THE INTER.
7417							
7418	041070	004737	041156	RK5111:	JSR	PC,RK5YY	
7419	041074	000237			SPL	7	
7420							
7421	041076	104450			CALRK5		
7422	041100	107			.BYTE	107	
7423	041101	000		RK5112:	.BYTE	0	
7424	041102	000000		RK5113:	.WORD	0	
7425	041104	000000		RK5114:	.WORD	0	
7426	041106	000000		RK5115:	.WORD	0	
7427	041110	000000		RK5116:	.WORD	0	
7428	041112	004000			.WORD	4000	
7429	041114	041132			.WORD	RK5SS	

```

7430 041116 000240      NOP
7431 041120 004737 041156  JSR    PC,RK5YY
7432 041124 005066 000002  CLR    2(SP)
7433 041130 000002      RTI
7434
7435 041132 000240      RK5SS: NOP
7436 041134 004737 041156  JSR    PC,RK5YY      ;SEE IF ANY ERRORS OCCURRED.
7437
7438 041140 005337 036076  DEC    RK5CR      ;DECRIEMENT THE PASS COUNT.
7439 041144 001001      BNE    RK5XX      ;IF NOT DONE CONTINUE.
7440 041146 000002      RTI              ;IF DONE GET OUT.
7441
7442 041150 000240      RK5XX: NOP
7443 041152 000137 040446  JMP    RK5AA      ;RESTART.
7444
7445 041156 000240      RK5YY: NOP
7446 041160 005737 060330  TST    RK5ER1     ;SEE IF ANY ERRORS OCCURRED.
7447 041164 001420      BEQ    RK5ZZ      ;IF NOT THEN RETURN TO CALL.
7448
7449 041166 000237      SPL    7
7450 041170 005037 036076  CLR    RK5CR      ;IF YES THEN CLEAR THE PASS COUNT.
7451 041174 013737 060332 001630  MOV    RK5ER2,$TMP1 ;AND MAKE AN ERROR CALL.
7452 041202 013737 060336 001634  MOV    RK5ER4,$TMP3
7453 041210 013737 060334 001632  MOV    RK5ER3,$TMP2
7454 041216 104160      ERROR 160
7455 041220 000230      SPL    0
7456 041222 005726      TST    (SP)+
7457 041224 000002      RTI              ;RETURN TO WAIT LOOP, DROPPING THIS DEVICE
7458                          ;FROM THE TEST.
7459
7460 041226 000207      RK5ZZ: RTS    PC      ;THERE WERE NO ERRORS.
7461
7462 041230 000000      DRK5T1: .WORD 0
7463 041232 000000      DRK5T2: .WORD 0
7464
7465
7466
7467
7468                          ;THIS IS THE UBE DRIVER ROUTINE USED IN THE CACHE I/O ARBITRATION
7469                          ;TEST.
7469 041234 012737 050050 041602  DRUBE: MOV    #50050,DUBET1 ;INITIALIZE THE RANDOM DATA
7470 041242 012737 060060 041604      MOV    #60060,DUBET2 ;GENERATER.
7471 041250 012737 070070 041410      MOV    #70070,UBEAA3
7472
7473 041256 104412      UBEAA: SAVREG
7474 041260 004737 042066  JSR    PC,GETBUF   ;PICK UP A MEMORY BUFFER
7475 041264 036136      .WORD UBERB
7476
7477 041266 013701 036136  MOV    UBERB,R1    ;COMPUTE THE MEMORY ADDRESS.
7478 041272 005000      CLR    R0
7479 041274 073027 000014  ASHC  #12,,R0
7480 041300 010137 041412  MOV    R1,UBEAA1
7481 041304 010137 041436  MOV    R1,UBEDD
7482 041310 010137 041476  MOV    R1,UBEII
7483 041314 010037 041414  MOV    R0,UBEAA2
7484 041320 010037 041440  MOV    R0,UBEEE
7485 041324 010037 041500  MOV    R0,UBEJJ
    
```

```

7486
7487 041330 000237          SPL      7
7488 041332 013737 041602 053106      MOV     DUBET1,$HINUM
7489 041340 013737 041604 053110      MOV     DUBET2,$LONUM
7490 041346 004737 053010          JSR     PC,$RAND
7491 041352 013737 053106 041602      MOV     $HINUM,DUBET1
7492 041360 013737 053110 041604      MOV     $LONUM,DUBET2
7493 041366 000230          SPL      0
7494
7495 041370 013703 041602          MOV     DUBET1,R3          ;SET THE UNIBUS TESTER DATA REG.
7496 041374 010337 041474          MOV     R3,UBEHH
7497 041400 010337 041434          MOV     R3,UBECCC
7498
7499 041404 104414          RESREG
7500
7501 041406 104440          WRRAND          ;FILL THE MEMORY BUFFER WITH
7502 041410 000000          UBEAA3: .WORD 0          ;RANDOM DATA.
7503 041412 000000          UBEAA1: .WORD 0
7504 041414 000000          UBEAA2: .WORD 0
7505 041416 004000          .WORD 4000
7506 041420 005237 041410          INC     UBEAA3
7507
7508 041424 000237          SPL      7
7509 041426 104452          CALUBE          ;DO A READ MEMORY FUNCTION.
7510 041430 042543          .WORD 42543
7511 041432 000000          UBEBB: .WORD 0
7512 041434 000000          UBECCC: .WORD 0
7513 041436 000000          UBEDD: .WORD 0
7514 041440 000000          UBEEF: .WORD 0
7515 041442 010000          .WORD 10000
    
```



```

7516 041444 041460          .WORD  UBEFF
7517
7518 041446 004737 041540    JSR    PC,UBEYY
7519 041452 005066 000002    CLR    2(SP)
7520 041456 000002          RTI    ;GO DO SOMETHING ELSE WHILE
7521                                     ;WAITING FOR INTERRUPT.
7522 041460 004737 041540    UBEFF: JSR    PC,UBEYY
7523
7524 041464 000237          SPL    7
7525 041466 104452          CALUBE ;DO A WRITE MEMORY FUNCTION.
7526 041470 042543          .WORD  42543
7527 041472 000000    UBEGG: .WORD  0
7528 041474 000000    UBEHH: .WORD  0
7529 041476 000000    UBEII: .WORD  0
7530 041500 000000    UBEJJ: .WORD  0
7531 041502 010000          .WORD  10000
7532 041504 041520          .WORD  UBEKK
7533
7534 041506 004737 041540    JSR    PC,UBEYY
7535 041512 005066 000002    CLR    2(SP)
7536 041516 000002          RTI    ;GO DO SOMETHING ELSE WHILE
7537                                     ;WAITING FOR THE INTERRUPT.
7538 041520 004737 041540    UBEKK: JSR    PC,UBEYY
7539
7540 041524 005337 036100          DEC    UBECR ;DECREMENT THE PASS COUNT.
7541 041530 001001          BNE    UBECC ;BR IF NOT DONE
7542
7543 041532 000002          RTI    ;IF DONE RETURN.
7544 041534 000137 041256    UBELL: JMP    UBEAA ;IF NOT DONE DO ANOTHER PASS.
7545
7546 041540 005737 061344    UBEYY: TST   UBEFR1 ;WERE THERE ANY ERRORS?
7547 041544 001415          BEQ    UBEZZ ;BR IF NO.
7548
7549 041546 000237          SPL    7 ;IF THERE WERE REPORT DEVICE FAILURE.
7550 041550 005037 036100          CLR    UBECC
7551 041554 013737 061346 001630    MOV    UBEER2,$TMP1
7552 041562 013737 061350 001632    MOV    UBEER3,$TMP2
7553 041570 104161          ERROR  161
7554 041572 005726          TST   (SP)+
7555 041574 000230          SPL    0
7556 041576 000002          RTI    ;RETURN WITH THIS DRIVER LOCKED OUT.
7557 041600 000207    UBEZZ: RTS    PC ;NO ERRORS CONTINUE.
7558
7559 041602 000000          DUBET1: .WORD  0
7560 041604 000000          DUBET2: .WORD  0
7561
7562
7563
7564 ;THIS ROUTINE IS USED TO GENERATE A BUFFER FULL OF RANDOM DATA.
7565 ;IT IS CALLED USING THE TRAP TABLE CALL:
7566 :      WRRAND
7567 :      .WORD  HIGHNUM
7568 :      .WORD  LOADRS
7569 :      .WORD  HIGHADRS
7570 :      .WORD  WORDCOUNT
7571 :RET:
    
```

```

7572
7573
7574
7575
7576
7577 041606 000237
7578 041610 011637 041750
7579 041614 062716 000010
7580 041620 104412
7581 041622 013700 041750
7582 041626 012001
7583 041630 012002
7584 041632 012003
7585 041634 012004
7586 041636 010237 041746
7587 041642 010337 041744
7588 041646 010137 053106
7589 041652 005037 053110
7590
7591 041656 013702 041744
7592 041662 013703 041746
7593 041666 073227 177772
7594 041672 010337 172354
7595 041676 013702 041746
7596 041702 042702 177700
7597 041706 062702 140000
7598 041712 004737 053010
7599 041716 013712 053106
7600 041722 062737 000002 041746
7601 041730 005537 041744
7602 041734 077430
7603
7604 041736 000230
7605 041740 104414
7606 041742 000002
7607
7608 041744 000000
7609 041746 000000
7610 041750 000000
7611
7612
7613 041752 012700 036126
7614 041756 012701 000005
7615
7616 041762 005020
7617 041764 077102
7618 041766 104424
7619 041770 000000
7620 041772 000000
7621 041774 062737 000002 041770
7622 042002 005537 041772
7623 042006 013700 041772
7624 042012 013701 041770
7625 042016 073027 177764
7626 042022 010137 042054
7627 042026 162701 000011

```

```

;WHERE HIGHNUM IS THE HIGH ORDER PART OF THE NUMBER USED TP PRIME THE
;RANDOM NUMBER GENERATER. THE LOW ORDER PART OF THAT NUMBER IS ASSUMED
;TO BE ZERO. LOADRS AND HIGHADRS IS THE 22 BIT ADDRESS OF THE BUFFER
;IN MEMORY WHICH WILL BE FILLED. WORDCOUNT IS THE NUMBER OF LOCATIONS
;TO BE WRITTEN.

```

```

RANDWR: SPL 7
MOV (SP),RANDTP
ADD #10,(SP)
SAVREG
MOV RANDTP,R0
MOV (R0)+,R1
MOV (R0)+,R2
MOV (R0)+,R3
MOV (R0)+,R4
MOV R2,RLWT
MOV R3,RHWT
MOV R1,$HINUM
CLR $LONUM

```

```

1$: MOV RHWT,R2 ;COMPUTE THE VIRTUAL ADDRESS OF THE BUFFER WORD.
MOV RLWT,R3
ASHC #-6,R2
MOV R3,$WKIPAR6
MOV RLWT,R2
BIC #177700,R2
ADD #140000,R2
JSR PC,$RAND
MOV $HINUM,(R2)
ADD #2,RLWT
ADC RHWT
SOB R4,1$

```

```

SPL 0
RESREG
RTI
RHWT: .WORD 0
RLWT: .WORD 0
RANDTP: .WORD 0

```

```

;THIS ROUTINE IS USED TO INITIALIZE THE GET BUFFER ROUTINE.
GTBINT: MOV #RS4RB,R0 ;CLEAR ALL THE BUFFER POINTERS.
MOV #5,R1

```

```

1$: CLR (R0)+
SOB R1,1$
SIZE ;COMPUTE THE SIZE OF MEMORY.

```

```

GTBILO: .WORD 0
GTBIHI: .WORD 0
ADD #2,GTBILO
ADC GTBIHI
MOV GTBIHI,R0 ;COMPUTE THE 2K BLOCK SIZE OF MEMORY.
MOV GTBILO,R1
ASHC #-12,R0
MOV R1,GTMSIZ
SUB #11,R1

```

```

7628 042032 010137 042056
7629 042036 012737 123456 042060
7630 042044 012737 123456 042062
7631 042052 000207
7632
7633 042054 000000
7634 042056 000000
7635 042060 000000
7636 042062 000000
7637 042064 000000
7638
7639
7640
7641
7642 042066 000237
7643 042070 011637 042064
7644 042074 062716 000002
7645
7646 042100 104412
7647 042102 013737 042060 053110 1$:
7648 042110 013737 042062 053106
7649 042116 004737 053010
7650 042122 013737 053110 042060
7651 042130 013701 053106
7652 042134 010137 042062
7653 042140 005000
7654 042142 071037 042056
7655
7656 042146 012702 036126
7657 042152 012703 000005
7658 042156 062701 000011
7659
7660 042162 020122
7661 042164 001746
7662 042166 077303
7663
7664 042170 017704 177670
7665 042174 010114
7666 042176 104414
7667 042200 000230
7668 042202 000207
7669
7670
7671 042204 104416
7672
7673
7674
7675
7676
7677
7678
7679
7680
7681
7682
7683

```

```

MOV R1,AVMBL
MOV #123456,GTRNL
MOV #123456,GTRNH
RTS PC

GTMSIZ: .WORD 0
AVMBL: .WORD 0
GTRNL: .WORD 0
GTRNH: .WORD 0
GETMP1: .WORD 0

```

```

;THIS ROUTINE IS CALLED TO ALLOCATE A MEMORY BUFFER OF 2K WORDS LENGTH.
;IT IS CALLED USING A JSR PC INSTRUCTION FOLLOWED BY THE TABLE ENTRY
;OF RS4RB TO BE UPDATED.
GETBUF: SPL 7 ;LOCK OUT INTERRUPTS.
MOV (SP),GETMP1
ADD #2,(SP) ;PICK UP A POINTER TO THE ARGUMENT
;AND UPDATE THE RETURN ADDRESS.

```

```

SAVREG
1$: MOV GTRNL,$LONUM
MOV GTRNH,$HINUM
JSR PC,$RAND
MOV $LONUM,GTRNL
MOV $HINUM,R1
MOV R1,GTRNH
CLR R0
DIV AVMBL,R0

MOV #RS4RB,R2 ;SEE IF THIS AREA IS ALREADY IN USE.
MOV #5,R3
ADD #11,R1

2$: CMP R1,(R2)+
BEQ 1$ ;IF IT IS THEN TRY AGAIN.
SOB R3,2$

MOV @GETMP1,R4 ;OTHERWISE GIVE THIS BUFFER TO THE DRIVER.
MOV R1,(R4)
RESREG
SPL 0
RTS PC

```

INDONE: RSET

```

;*****
;*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.
;*

```

```

7684 .....
7685 042206 000004 TST36: SCOPE
7686
7687 042210 012737 047736 054230 MOV #KT,SKAD ;SET THE SKAD REGISTER
7688 ;IN CASE THE TEST ABORTS.
7689 042216 104416 RSET
7690 042220 113737 001502 001626 MOVB $TSTNM,$TMP0
7691 042226 004737 055456 JSR PC,SIZDEV ;DETERMINE WHAT DEVICES ARE AVAILABLE.
7692 042232 113737 056050 042746 MOVB RS4DFL,RS4FT
7693 042240 113737 056051 042747 MOVB RP4DFL,RP4FT
7694 042246 113737 056052 042750 MOVB RH4DFL,RH4FT
7695
7696 042254 000137 043064 NN1: JMP NNDEV ;GO COMPUTE THE DRIVE NUMBERS.
7697
7698 042260 005037 042744 NN2: CLR NNGRPF ;FLAG WHICH DESIGNATES WHICH GOUP IS BEING
7699 042264 012737 000044 042742 MOV #S1M0,NNGRM ;TESTED ON THIS PASS.
7700 042272 012737 000030 042740 MOV #SOM1,NNGRS ;TEST GROUP ZERO FIRST.
7701
7702 042300 004737 042754 NN3: JSR PC,NNSTUP ;GO MAKE THE TEST ADDRESSES HITS
7703 042304 004777 000426 JSR PC,@NNUD ;USE THE FIRST DEVICE.
7704
7705
7706 042310 012700 140000 MOV #TESTR1,R0
7707 042314 012701 000400 MOV #256.,R1 ;MAKE SURE THOSE ADDRESSES ARE MISSES.
7708
7709 042320 005710 1$: TST (R0)
7710 042322 032737 000010 177752 BIT #10,@HITMIS
7711 042330 001430 BEQ 2$
7712
7713 042332 013737 042744 001630 MOV NNGRPF,$TMP1 ;GOT A HIT REPORT FAILURE.
7714 042340 010037 001632 MOV R0,$TMP2
7715 042344 005037 001634 CLR $TMP3
7716 042350 023727 042736 042552 CMP NNUD,#NNRS4 ;WAS THE RS4 DOING THE TRANSFER?
7717 042356 001003 BNE 11$ ;BRANCH IF NOT.
7718 042360 104151 ERROR 151
7719 042362 000137 042420 JMP NN5
7720 042366 023727 042736 042644 11$: CMP NNUD,#NNRP4 ;WAS IT THE RP4?
7721 042374 001003 BNE 12$
7722 042376 104152 ERROR 152
7723 042400 000137 042420 JMP NN5
7724 042404 104153 12$: ERROR 153
7725 042406 000137 042420 JMP NN5
7726
7727 042412 062700 000004 2$: ADD #4,R0
7728 042416 077140 SOB R1,1$
7729
7730 042420 005237 042744 NN5: INC NNGRPF ;TESTED BOTH GROUPS?
7731 042424 022737 000002 042744 CMP #2,NNGRPF
7732 042432 001410 BEQ NN6 ;BRANCH IF YES.
7733 042434 012737 000044 042740 MOV #S1M0,NNGRS ;IF NOT GO BACK AND TEST GROUP ONE.
7734 042442 012737 000030 042742 MOV #SOM1,NNGRM
7735 042450 000137 042300 JMP NN3
7736
7737 042454 000137 043332 NN6: JMP NNDONE
7738
7739 042460 104446 NNRH4: CALRH4 ;THIS IS THE CALL TO READ THE MASS BUS TESTER.

```

```

7740 042462 071
7741 042463 000
7742 042464 052525
7743 042466 000000
7744 042470 140000
7745 042472 000000
7746 042474 001000
7747 042476 042510
7748
7749 042500 005737 062076 1$: TST RH4ER1 ;ANY DEVICE ERRORS?
7750 042504 100401 BMI 2$ ;BRANCH IF YES.
7751 042506 000207 RTS PC ;IF NOT RETURN.
7752
7753 042510 013737 062100 001630 2$: MOV RH4ER2,$TMP1 ;REPORT DEVICE ERROR.
7754 042516 013737 062102 001632 MOV RH4ER3,$TMP2
7755 042524 013737 062104 001634 MOV RH4ER4,$TMP3
7756 042532 005726 TST (SP)+
7757 042534 104156 ERROR 156
7758 042536 105037 056052 CLRB RH4DFL
7759 042542 105037 042750 CLRB RH4FT
7760 042546 000137 042254 JMP NN1
7761
7762 042552 104442 NNRS4: CALRS4 ;THIS IS A CAL TO DO AN RS4 READ.
7763 042554 071 .BYTE 71
7764 042555 000 NNRS4U: .BYTE 0
7765 042556 000000 .WORD 0
7766 042560 000000 .WORD 0
7767 042562 140000 .WORD TESTR1
7768 042564 000000 .WORD 0
7769 042566 001000 .WORD 512.
7770 042570 042602 .WORD 2$
7771
7772 042572 005737 057374 1$: TST RS4ER1 ;SEE IF THERE WERE DEVICE ERRORS.
7773 042576 100401 BMI 2$ ;BR IF YES.
7774 042600 000207 RTS PC
7775
7776 042602 013737 057376 001630 2$: MOV RS4ER2,$TMP1
7777 042610 013737 057400 001632 MOV RS4ER3,$TMP2
7778 042616 013737 057402 001634 MOV RS4ER4,$TMP3
7779 042624 005726 TST (SP)+
7780 042626 104154 ERROR 154
7781 042630 105037 056050 CLRB RS4DFL
7782 042634 105037 042746 CLRB RS4FT
7783 042640 000137 042254 JMP NN1
7784
7785 042644 104444 NNRP4: CALRP4 ;THIS IS A CALL TO DO AN RP4 READ.
7786 042646 071 .BYTE 71
7787 042647 000 NNRP4U: .BYTE 0
7788 042650 000000 .WORD 0
7789 042652 000000 .WORD 0
7790 042654 140000 .WORD TESTR1
7791 042656 000000 .WORD 0
7792 042660 001000 .WORD 512.
7793 042662 042674 .WORD 2$
7794
7795 042664 005737 056424 1$: TST RP4ER1 ;WERE THERE ANY DEVICE ERRORS?

```

```

7796 042670 100401      BMI      2$
7797 042672 000207      RTS      PC
7798
7799 042674 013737 056426 001630 2$:      MOV      RP4ER2,$TMP1
7800 042702 013737 056430 001632      MOV      RP4ER3,$TMP2
7801 042710 013737 056432 001634      MOV      RP4ER4,$TMP3
7802 042716 005726      TST      (SP)+
7803 042720 104155      ERROR   155
7804 042722 105037 056051      CLRB    RP4DFL
7805 042726 105037 042747      CLRB    RP4FT
7806 042732 000137 042254      JMP     NN1
7807
7808 042736 000000      NNUD:   .WORD 0
7809
7810 042740 000000      NNGRS:  .WORD 0
7811 042742 000000      NNGRM:  .WORD 0
7812 042744 000000      NNGRPF: .WORD 0
7813
7814      ;THIS ROUTINE IS CALLED TO MAKE THE ADDRESSES IN TESTR1
7815      ;HITS PRIOR TO CALLING FOR THE MB DEVICE TO DO TRANSFERS.
7816 042746      000      RS4FT:  .BYTE 0
7817 042747      000      RP4FT:  .BYTE 0
7818 042750      000      RH4FT:  .BYTE 0
7819 042751      000      RK5FT:  .BYTE 0
7820 042752      000      UBFT:   .BYTE 0
7821      042754      .EVEN
7822
7823 042754 104412      NNSTUP: SAVREG
7824 042756 012700 042754      MOV     #NNSTUP,R0      ;MAKE THIS CODE HITS IN THE
7825 042762 012701 001000      MOV     #512.,R1      ;GROUP NOT BEING TESTED.
7826 042766 012702 142000      MOV     #TESTR2,R2
7827
7828 042772 013737 042742 177746 1$:      MOV     NNGRM,@CONTRL
7829 043000 005720      TST     (R0)+
7830 043002 013737 042740 177746      MOV     NNGRS,@CONTRL
7831 043010 005722      TST     (R2)+
7832 043012 077111      SOB     R1,1$
7833
7834 043014 013700 042740      2$:      MOV     NNGRS,R0
7835 043020 042700 000014      BIC     #14,R0
7836 043024 010037 177746      MOV     R0,@CONTRL
7837 043030 012701 140000      MOV     #TESTR1,R1
7838 043034 012702 001000      MOV     #512.,R2
7839 043040 005721      3$:      TST     (R1)+
7840 043042 077202      SOB     R2,3$
7841 043044 013700 042742      MOV     NNGRM,R0
7842 043050 042700 000014      BIC     #14,R0
7843 043054 010037 177746      MOV     R0,@CONTRL
7844 043060 104414      RESREG
7845 043062 000207      RTS     PC
7846
7847
7848      ;SEE WHAT DEVICE TO USE NEXT.
7849 043064 000240      NNDEV:  NOP
7850 043066 000240      NOP
7851 043070 005037 042736      CLR     NNUD

```

```

7852 043074 113700 042746      MOVB   RS4FT,R0      ;IS THERE AN RS4 DRIVE.
7853 043100 001430      BEQ    NND2          ;BR IS NOT
7854
7855 043102 000240      NND0:  NOP
7856 043104 012701 000001      MOV    #1,R1        ;FIND OUT WHAT DRIVE NUMBER IT IS.
7857 043110 012737 042552 042736      MOV    #NRS4,NNUD
7858 043116 005002      CLR    R2
7859 043120 012703 000010      MOV    #10,R3
7860 043124 000240      1$:   NOP
7861 043126 030100      BIT    R1,R0
7862 043130 001406      BEQ    2$
7863 043132 140137 042746      BICB   R1,RS4FT     ;FOUND IT.
7864 043136 110237 042555      MOVB   R2,NNRS4U
7865 043142 000137 042260      JMP    NN2
7866 043146 005202      2$:   INC    R2
7867 043150 006301      ASI    R1
7868 043152 077314      SOB    R3,1$       ;KEEP LOOKING.
7869
7870 043154 104000      ERROR  0
7871 043156 105037 042746      CLRB   RS4FT
7872
7873 043162 000240      NND2:  NOP
7874 043164 113700 042747      MOVB   RP4FT,R0     ;IS THERE AN RP04 DRIVE.
7875 043170 001426      BEQ    NND3         ;BR IF NO
7876 043172 012701 000001      MOV    #1,R1
7877 043176 012737 042644 042736      MOV    #NRP4,NNUD
7878 043204 005002      CLR    R2
7879 043206 000010      MOV    #10,R3
7880 043212 030100      1$:   BIT    R1,R0
7881 043214 001406      BEQ    2$
7882 043216 140137 042746      BICB   R1,RS4FT
7883 043222 110237 042647      MOVB   R2,NNRP4U
7884 043226 000137 042260      JMP    NN2
7885 043232 005202      2$:   INC    R2
7886 043234 006301      ASL    R1
7887 043236 077313      SOB    R3,1$
7888 043240 104000      ERROR  0
7889 043242 105037 042747      CLRB   RP4FT
7890
7891 043246 000240      NND3:  NOP
7892 043250 113700 042750      MOVB   RH4FT,R0     ;IS THERE A MASS BUS TESTER.
7893 043254 001426      BEQ    NNDONE
7894 043256 012701 000001      MOV    #1,R1
7895 043262 012737 042460 042736      MOV    #NRRH4,NNUD
7896 043270 005002      CLR    R2
7897 043272 012703 000010      MOV    #10,R3
7898 043276 030100      1$:   BIT    R1,R0
7899 043300 001406      BEQ    2$
7900 043302 140137 042750      BICB   R1,RH4FT
7901 043306 110237 042463      MOVB   R2,NNRH4U
7902 043312 000137 042260      JMP    NN2
7903 043316 005202      2$:   INC    R2
7904 043320 006301      ASL    R1
7905 043322 077313      SOB    R3,1$
7906 043324 104000      ERROR  0
7907 043326 105037 042750      CLRB   RH4FT

```

```

7908 043332 104416          NNDONE: RSET
7909
7910
7911
7912 043334 105737 001714      TSTB   KB11CM           ;11/74      (KB11CM)?
7913 043340 001005          BNE    1$              ;BRANCH IF YES
7914 043342 105737 001713      TSTB   KB11EM           ;KB11-EM?
7915 043346 001002          BNE    1$              ;BR IF YES
7916 043350 000137 047736      JMP    KT              ;GO TO KT IF NO
7917 043354          1$:                   ;ENTER HERE IF KB11-E
7918
7919
7920
7921
7922
7923
7924
7925
7926 043354 000004          *****
7927 043356 005037 177746      :*TEST 37      CHECK IVSS, VSIU BITS
7928 043362 005737 177746          :THIS TEST CHECKS THAT THE IVSS AND VSIU BITS OF THE CACHE
7929 043366 001404          :CONTROL REGISTER CAN BE SET AND CLEARED. VCIP IS ALSO
7930 043370 013737 177746 001556      :CHECKED.
7931 043376 104055          :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
7932
7933
7934
7935
7936
7937
7938
7939
7940
7941
7942
7943
7944
7945
7946
7947
7948
7949
7950
7951
7952
7953
7954
7955
7956
7957
7958
7959
7960
7961
7962
7963
    TST37: SCOPE
            CLR    @#CONTRL
            TST    @#CONTRL
            BEQ    1$
            MOV    @#CONTRL,$REGO
            ERROR  55      ;CCR COULD NOT BE CLEARED
    1$:     MOV    #IVSS,@#CONTRL
            CMP    #IVSS,@#CONTRL
            BEQ    2$
            MOV    @#CONTRL,$REGO
            ERROR  56      ;IVSS COULD NOT BE SET
    2$:     BIC    #IVSS,@#CONTRL
            BEQ    3$
            MOV    @#CONTRL,$REGO
            ERROR  57      ;IVSS COULD NOT BE CLEARED
    3$:     MOV    #VSIU,@#CONTRL
            BIT    #VSIU,@#CONTRL
            BNE    4$
            MOV    @#CONTRL,$REGO
            ERROR  60      ;VSIU COULD NOT BE SET
    4$:     MOV    #50,R0      ;WAIT FOR VCIP TO CLEAR
            BIT    #VCIP,@#CONTRL
            BEQ    5$
            SOB   R0,4$
            MOV    @#CONTRL,$REGO
            ERROR  61      ;VCIP DID NOT CLEAR WITHIN SOME
                                ;SOME TIME AFTER VSIU WAS SET
    5$:     BIC    #VSIU,@#CONTRL
            BIT    #VSIU,@#CONTRL
            BEQ    6$
            MOV    @#CONTRL,$REGO
            ERROR  62      ;VSIU COULD NOT BE CLEARED
    6$:     BIT    #VCIP,@#CONTRL
            BNE    6$
    
```



```

7964
7965
7966
7967
7968
7969
7970
7971 043560 000004
7972
7973 043562 012737 040000 177746
7974 043570 032737 020000 177746
7975 043576 001404
7976 043600 013737 177746 001556
7977 043606 104062
7978
7979 043610 032737 010000 177746 1$:
7980 043616 001374
7981 043620 052737 020000 177746
7982 043626 032737 020000 177746
7983 043634 001004
7984 043636 013737 177746 001556
7985 043644 104060
7986
7987 043646 042737 020000 177746 2$:
7988 043654 032737 020000 177746
7989 043662 001404
7990 043664 013737 177746 001556
7991 043672 104062
7992
7993
7994
7995
7996
7997
7998
7999
8000
8001 043674 000004
8002 043676 012737 000400 177746
8003 043704 000240
8004 043706 012700 000062
8005 043712 032737 010000 177746
8006 043720 001004
8007 043722 013737 177746 001556
8008 043730 104063
8009
8010 043732 032737 010000 177746 1$:
8011 043740 001405
8012 043742 077005
8013 043744 013737 177746 001556
8014 043752 104061
8015
8016
8017 043754
8018
8019

```

```

*****
*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.
;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
*****
TST40: SCOPE
MOV #IVSS,@CONTR
BIT #VSIU,@CONTR
BEQ 1$
MOV @CONTR,$REGO
ERROR 62 ;VALID STORE IN USE, BIT 13,
;COULD NOT BE CLEARED IN CCR
1$: BIC #VCIP,@CONTR
BNE 1$
BIS #VSIU,@CONTR
BIT #VSIU,@CONTR
BNE 2$
MOV @CONTR,$REGO
ERROR 60 ;VSIU (BIT 13) COULD NOT BE SET
;IN CCR (IVSS WAS ALREADY SET).
2$: BIC #VSIU,@CONTR
BIT #VSIU,@CONTR
BEQ TST41 ;;EXIT
MOV @CONTR,$REGO
ERROR 62 ;VSIU COULD NOT BE CLEARED IN CCR
;IVSS WAS ALREADY SET.
*****
*TEST 41 CHECK VCIP SETS WHEN CF IS SET
;THIS TEST CHECKS THAT THE VCIP SETS WHEN CACHE-FLUSH IS
;DONE AND IT CLEARS OUT WITHIN A CERTAIN TIME AFTER
;THE FLUSH OF VALID STORE IS OVER
;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
*****
TST41: SCOPE
MOV #FCAC,@CONTR;FLUSH CACHE
NOP
MOV #50,R0
BIT #VCIP,@CONTR
BNE 1$
MOV @CONTR,$REGO
ERROR 63 ;VCIP DID NOT SET WHEN CACHE
;FLUSH WAS ISSUED
1$: BIT #VCIP,@CONTR ;WAIT FOR VCIP TO CLEAR
BEQ 2$
SOB R0,1$
MOV @CONTR,$REGO
ERROR 61 ;VCIP DID NOT CLEAR WITHIN A
;CERTAIN TIME AFTER CACHE FLUSH
;WAS DONE
2$:
*****

```

```

8020      ;*TEST 42      CHECK CACHE FLUSH & VALID STORE SWITCHING
8021      ;THIS TEST CHECKS THAT WHEN A CACHE FLUSH IS DONE
8022      ;BY SETTING CF IN CCR, THE VALID STORE IN USE
8023      ;(VSIU) SURTCHES. VALID STORE SWITCHING FROM STORE-A
8024      ;TO STORE-B AND VICE-VERSA IS CHECKED
8025      ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8026      ;*****

```

```

8027 043754 000004      ;TST42: SCOPE
8028 043756 005037 177746      CLR      @#CONTRL
8029 043762 032737 010000 177746 1$:      BIT      #VCIP, @#CONTRL
8030 043770 001374      BNE      1$
8031 043772 012737 000400 177746      MOV      #FCAC, @#CONTRL ;FLUSH CACHE
8032 044000 032737 020000 177746      BIT      #VSIU, @#CONTRL
8033 044006 001004      BNE      2$
8034 044010 013737 177746 001556      MOV      @#CONTRL, $REGO
8035 044016 104064      ERROR   64      ;VSIU DID NOT SWITCH FROM 0 TO 1
8036      ;WHEN CACHE FLUSH WAS SET
8037 044020 032737 010000 177746 2$:      BIT      #VCIP, @#CONTRL
8038 044026 001374      BNE      2$
8039 044030 012737 000400 177746      MOV      #FCAC, @#CONTRL
8040 044036 032737 020000 177746      BIT      #VSIU, @#CONTRL
8041 044044 001404      BEQ      3$
8042 044046 013737 177746 001556      MOV      @#CONTRL, $REGO
8043 044054 104064      ERROR   64      ;VSIU DID NOT SWITCH FROM 1 TO 0 WHEN
8044      ;FLUSH-CACHE WAS SET IN CCR
8045 044056 032737 010000 177746 3$:      BIT      #VCIP, @#CONTRL
8046 044064 001374      BNE      3$
8047
8048
8049

```

```

8050      ;*****
8051      ;*TEST 43      CHECK IVSS INHIBITS SWITCHING OF VALID STORE IN USE
8052      ;THIS TEST CHECKS THAT WHEN "INHIBIT VALID STORE SWITCHING"
8053      ;(IVSS) IS SET AND FLUSH-CACHE BIT IS SET, THE
8054      ;VALID STORE IN USE DOES NOT SWITCH
8055      ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8056      ;*****

```

```

8056 044066 000004      ;TST43: SCOPE
8057 044070 005037 177746      CLR      @#CONTRL
8058 044074 032737 010000 177746 1$:      BIT      #VCIP, @#CONTRL
8059 044102 001374      BNE      1$
8060 044104 012737 040000 177746      MOV      #IVSS, @#CONTRL ;SET IVSS
8061 044112 052737 000400 177746      BIS      #FCAC, @#CONTRL ;FLUSH CACHE
8062 044120 032737 020000 177746      BIT      #VSIU, @#CONTRL
8063 044126 001404      BEQ      2$      ;CHECK VSIU DID NOT SWITCH
8064 044130 013737 177746 001556      MOV      @#CONTRL, $REGO
8065 044136 104065      ERROR   65      ;VSIU SWITCHED, WHEN IVSS
8066      ;WAS SET AND CACHE FLUSH
8067      ;WAS DONE, IT SHOULD NOT SWITCH
8068 044140 032737 010000 177746 2$:      BIT      #VCIP, @#CONTRL
8069 044146 001374      BNE      2$
8070 044150 052737 020000 177746      BIS      #VSIU, @#CONTRL
8071 044156 032737 010000 177746 3$:      BIT      #VCIP, @#CONTRL
8072 044164 001374      BNE      3$
8073 044166 052737 000400 177746      BIS      #FCAC, @#CONTRL
8074 044174 032737 020000 177746      BIT      #VSIU, @#CONTRL ;CHECK VSIU DID NOT SWITCH
8075 044202 001004      BNE      4$

```

8076 044204 013737 177746 001556
8077 044212 104065
8078
8079
8080 044214 032737 010000 177746 4\$:
8081 044222 001374
8082
8083
8084
8085
8086
8087
8088
8089
8090
8091
8092
8093
8094
8095
8096
8097
8098
8099
8100
8101
8102
8103 044224 000004
8104 044226 005005
8105 044230 010537 177746
8106 044234 032737 010000 177746
8107 044242 001374
8108 044244 012702 000034
8109 044250 012703 000054
8110 044254 050502
8111 044256 050503
8112 044260 012700 044226
8113 044264 012701 001000
8114 044270 010237 177746
8115 044274 005762 002000
8116 044300 010337 177746
8117 044304 005720
8118 044306 077110
8119 044310 012700 114704
8120 044314 012701 001000
8121 044320 010337 177746
8122 044324 042737 000014 177746
8123 044332 005720
8124 044334 077102
8125 044336 042737 000020 177746
8126 044344 052737 000040 177746
8127 044352 052737 000400 177746
8128 044360 013704 177746
8129 044364 074504
8130 044366 032704 020000
8131 044372 001004

MOV @#CONTRL,\$REGO
ERROR 65 ;VSIU SWITCHED, WHEN IVSS
; WAS SET AND CACHE FLUSH WAS
; DONE; IT SHOULD NOT SWITCH
BIT #VCIP, @#CONTRL
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.
: 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 WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
*****

```

```

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$

```

```

8132 044374 013737 177746 001556      MOV      @#CONTRL,$REG0
8133 044402 104064                      ERROR    64      ;VSIU DID NOT SWITCH WHEN
8134                                     ;CACHE-FLUSH WAS DONE
8135 044404 052737 000014 177746 3$:   BIS      #MOM1, @#CONTRL ;MAKE TEST-CODE HIT IN
8136 044412 012700 044226                      MOV      #VSG0, R0      ;GROUP 1
8137 044416 012701 001000                      MOV      #1000, R1
8138 044422 005720                      TST     (R0)+
8139 044424 077102                      SOB     R1, 4$
8140 044426 042737 000014 177746      BIC     #MOM1, @#CONTRL
8141 044434 012700 114704                      MOV      #TSTDAT, R0    ;REFERENCE TEST-DATA AND CHECK
8142 044440 012701 000400                      MOV      #400, R1      ;THAT IT IS A MISS. NOTE
8143 044444 005710                      TST     (R0)           ;SETTING CACHE-FLUSH SHOULD
8144 044446 032737 000010 177752      BIT     #10, @#HITMIS  ;HAVE INVALIDATED GROUP 0
8145 044454 001410                      BEQ     6$
8146 044456 013737 177746 001556      MOV      @#CONTRL,$REG0
8147 044464 005037 001560                      CLR     $REG1          ;GROUP NO.
8148 044470 010037 001562                      MOV     R0,$REG2      ;TEST DATA ADDRESS
8149 044474 104066                      ERROR    66      ;TEST-DATA WAS NOT A MISS.
8150                                     ;TEST DATA WAS MADE A HIT
8151                                     ;IN GROUP 0 AND THEN CACHE-
8152                                     ;FLUSH WAS DONE. CACHE-FLUSH
8153                                     ;SHOULD HAVE INVALIDATED GROUP
8154                                     ;0'S CACHED DATA, HENCE, THE
8155                                     ;TEST DATA REFERENCE SHOULD
8156                                     ;HAVE BEEN A MISS.
8157                                     ;PROBLE FAULURE
8158 044476 062700 000004          6$:   ADD     #4, R0      ;VALID STORE IS NOT BEING SWITCHED
8159 044502 077120                      SOB     R1, 5$      ;TO THE OTHER WHEN CACHE-FLUSH IS
8160                                     ;SET IN THE CCR
8161 044504 032737 010000 177746 7$:   BIT     #VCIP, @#CONTRL
8162 044512 001374                      BNE     7$
8163 044514 012700 020000                      MOV     #VSIU, R0      ;COMPLEMENT VSIU
8164 044520 074037 177746                      XOR     R0, @#CONTRL
8165 044524 032737 010000 177746 8$:   BIT     #VCIP, @#CONTRL
8166 044532 001374                      BNE     8$
8167 044534 052737 000014 177746      BIS     #MOM1, @#CONTRL ;MAKE TEST-CODE HIT IN
8168 044542 012700 044226                      MOV     #VSG0, R0      ;GROUP 1
8169 044546 012701 001000                      MOV     #1000, R1
8170 044552 005720                      TST     (R0)+
8171 044554 077102                      SOB     R1, 9$
8172 044556 042737 000014 177746      BIC     #MOM1, @#CONTRL
8173                                     ;THE ORIGINAL VALID STORE (WHICH
8174                                     ;WAS INVALIDATED BY CACHE FLUSH)
8175                                     ;IS IN USE AGAIN.
8176 044564 012700 114704                      MOV     #TSTDAT, R0
8177 044570 012701 000400                      MOV     #400, R1      ;REFERENCE THE TEST-DATA AND
8178                                     ;CHECK IT IS A MISS
8179 044574 005710                      TST     (R0)
8180 044576 032737 000010 177752      BIT     #10, @#HITMIS
8181 044604 001410                      BEQ     11$
8182 044606 013737 177746 001556      MOV     @#CONTRL,$REG0
8183 044614 005037 001560                      CLR     $REG1          ;GROUP NO.
8184 044620 010037 001562                      MOV     R0,$REG2      ;TEST DATA ADDRESS
8185 044624 104067                      ERROR    67      ;TEST-DATA REFERENCE WAS NOT A MISS (IN
8186                                     ;GROUP 0, ORIGINAL VALID STORE). CACHE-FLUSH
8187                                     ;DONE EARLIER ON THE ORIGINAL VALID STORE

```

```

8188                                     ;SHOULD HAVE RESULTED IN INVALIDATING
8189                                     ;THE VALID STORE, THUS RESULTING IN
8190                                     ;CACHE-MISS ON TEST DATA REFERENCE.
8191                                     ;PROBABLE FAILURE: VALID STORE IN USE IS NOT
8192                                     ;BEING INVALIDATED WHEN CACHE-FLUSH IS
8193                                     ;SET
8194 044626 062700 000004          11$:  ADD    #4, R0
8195 044632 077120                SOB    R1, 10$
8196 044634 012701 020000        MOV    #VSIU, R1
8197 044640 074105                XOR    R1, R5      ;TESTED VALID STORE B (1)?
8198 044642 001402                BEQ    TST45        ;;EXIT
8199 044644 000137 044230        JMP    VSGOA
8200
8201
8202                                     ;*****
8203 ;*TEST 45      CHECK VALID STORES (A&B) FOR GROUPS 0 & 1
8204 ;THIS TEST CHECKS THAT HIT CAN BE OBTAINED FROM BOTH GROUPS
8205 ; (0&1) OF THE CACHE, FROM EACH OF THE TWO VALID
8206 ; STORES (A&B) PER GROUP. THUS ALL 4 VALID STORES GET
8207 ; CHECKED.
8208 ; TEST-DATA (UNIQUE) IS MADE A HIT IN GROUP 0 USING
8209 ; THE FIRST VALID STORE A. TEST-CODE IS MADE A HIT IN THE
8210 ; GROUP NOT BEING TESTED. TEST-DATA IS READ BACK AND
8211 ; CHECKED FOR CORRECTNESS. IT IS ALSO CHECKED IF THE
8212 ; TEST-DATA REFERENCE WAS A HIT. THE TESTING IS
8213 ; REPEATED FOR VALID STORE B.
8214 ; THE ENTIRE TEST (ABOVE) IS REPEATED FOR GROUP 1.
8215 ; THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8216                                     ;*****
8217 044650 000004          TST45:  SCOPE
8218 044652 005002          G1GOV:  CLR    R2      ;VSIU HIT MASK
8219 044654 005005          CLR    R5      ;INITIALIZE COUNT DATA PATTERN TO BE USED
8220 044656 012700 000034    G1GOVA: MOV    #SOMOM1, R0
8221 044662 012701 000054    MOV    #S1MOM1, R1
8222 044666 010237 177746    G1GOVB: MOV    R2, @#CONTRL
8223 044672 032737 010000 177746 1$:  BIT    #VCIP, @#CONTRL
8224 044700 001374          BNE    1$
8225 044702 050200          BIS    R2, R0
8226 044704 050201          BIS    R2, R1
8227 044706 012703 044652    MOV    #G1GOV, R3      ;MAKE TEST-CODE HIT IN THE
8228 044712 012704 001000    MOV    #1000, R4      ;GROUP NOT BEING TESTED
8229 044716 010037 177746    2$:  MOV    R0, @#CONTRL
8230 044722 005763 002000    TST    2000 (R3)
8231 044726 010137 177746    MOV    R1, @#CONTRL
8232 044732 005723          TST    (R3)+
8233 044734 077410          SOB    R4, 2$
8234 044736 042700 000014    BIC    #MOM1, R0      ;WRITE COUNT PATTERN AND MAKE
8235 044742 042701 000014    BIC    #MOM1, R1      ;IT A HIT IN THE GROUP BEING
8236 044746 012703 114704    MOV    #TSTDAT, R3    ;TESTED.
8237 044752 012704 001000    MOV    #1000, R4      ;BIT 15 OF THE COUNT PATTERN INDICATES
8238                                     ;WHICH GROUP; BIT15-0, GROUP 0, ELSE 1
8239 044756 010037 177746    3$:  MOV    R0, @#CONTRL  ;BIT 14 OF THE COUNT PATTERN INDICATES
8240 044762 010513          MOV    R5, (R3)      ;WHICH VALID STORE, A (0) OR B (1)
8241 044764 005723          TST    (R3)+        ;MAKE IT A HIT
8242 044766 005205          INC    R5
8243 044770 077404          SOB    R4, 3$
    
```

```

8244 044772 010137 177746      MOV      R1, @#CONTRL
8245 044776 012703 114704      MOV      #TSTDAT, R3
8246 045002 012704 001000      MOV      #1000, R4
8247 045006 042705 001777      BIC      #1777, R5          ;INITIALIZE PATTERN TO BE CHECKED
8248 045012 011337 045166      MOV      (R3), TMP        ;READ THE TEST-DATA AND
8249 045016 032737 000020 177752 4$: BIT      #20, @#HITMIS    ;CHECK IF THE REFERENCE WAS
8250 045024 001016          BNE      5$              ;A HIT
8251 045026 013737 177746 001556      MOV      @#CONTRL, $REG0
8252 045034 005037 001560      CLR      $REG1          ;GROUP NO.
8253 045040 032705 100000      BIT      #BIT15, R5     ;WHICH GROUP?
8254 045044 001403          BEQ      8$
8255 045046 012737 000001 001560      MOV      #1, $REG1
8256 045054 010337 001562 8$: MOV      R3, $REG2
8257 045060 104070          ERROR   70              ;TEST DATA ADDRESS
8258          ;TEST-DATA REFERENCE WAS NOT A
8259          ;HIT, FROM THE GROUP AND
8260 045062 023705 045166 5$: CMP      TMP, R5
8261 045066 001410          BEQ      6$              ;DATA CORRECT?
8262 045070 010537 001556          MOV      R5, $REG0
8263 045074 013737 045166 001560      MOV      TMP, $REG1
8264 045102 010337 001562          MOV      R3, $REG2
8265 045106 104071          ERROR   71              ;READ INCORRECT DATA ON REFEREN
8266          ;-CING A CACHED LOCATION.
8267 045110 062703 000002 6$: ADD      #2, R3
8268 045114 005205          INC      R5
8269 045116 077443          SOB     R4, 4$
8270 045120 012704 020000      MOV      #VSIU, R4
8271 045124 074402          XOR     R4, R2          ;DONE VALID STORE B (1)?
8272 045126 001405          BEQ     7$              ;YES
8273 045130 052705 040000      BIS     #BIT14, R5     ;INDICATE VS-B IN DATA-PATTERN
8274 045134 042705 001777      BIC     #1777, R5
8275 045140 000646          BR     G1GOVA          ;CHECK GROUP, VS-B
8276 045142 032705 100000 7$: BIT     #BIT15, R5     ;DONE CHECKING GROUP 1?
8277 045146 001010          BNE     TST46          ;;EXIT
8278 045150 012700 000054      MOV     #SIMOM1, R0
8279 045154 012701 000034      MOV     #SOMOM1, R1
8280 045160 012705 100000      MOV     #BIT15, R5 ;INDICATE GROUP 1
8281 045164 000640          BR     G1GOVB
8282 045166 000000      TMP:   .WORD 0
    
```

```

8283
8284
8285 ;:*****
8286 ;*TEST 46 CHECK VALID STORES (A & B ) FOR GROUP 1
8287 ;THIS TEST CHECKS RTHE TWO VALID STORES (A&B) FOR GROUP 1
8288 ;OF THE CACHE. WHEN A CACHE-FLUSH IS ISSUED, THE CACHE
8289 ;SHOULD BE INVALIDATED BY SWITCHING THE VALID STORE
8290 ;IN USE.
8291 ;THE TEST-CODE IS MADE HIT IN GROUP 1 (WHICH IS NOT
8292 ;BEING TESTED). THE TEST DATA IS MADE HIT IN GROUP 0.
8293 ;FLUSH-CACHE HIT IS SET IN THE CCR. IT IS CHECKED THAT
8294 ;THE TEST-DATA WHICH WAS HIT (MADE PREVIOUSLY) IN
8295 ;GROUP 0 IS NO MORE A HIT, EACH LOCATION OF THE
8296 ;TEST-DATA BLOCK IS REFERENCED AND CHECKED IF
8297 ;IT WAS A MISS. OTHERWISE AN ERROR IS REPORTED. AS A
8298 ;RESULT OF THE CACHE FLUSH THE VALID STORE SHOULD
8299 ;HAVE SWITCHED FROM 0 TO 1. THEN THE VALID STORE
    
```

```

8300                                     :IS FORCED TO BE 0 AND THE TEST-DATA IS REFERENCED
8301                                     :AGAIN. IT IS CHECKED IF IT WAS A MISS.
8302                                     :THE WHOLE TEST IS REPEATED USING VALID-STORE
8303                                     :B (1).
8304                                     :THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8305                                     :*****
8306 045170 000004 TST46: SCOPE
8307 045172 005005 VSG1: CLR R5 ;R5, HIT MASK FOR VSIU
8308 045174 010537 177746 VSG1A: MOV R5, @#CONTRL
8309 045200 032737 010000 177746 BIT #VCIP, @#CONTRL
8310 045206 001374 BNE .-6
8311 045210 012702 000034 MOV #SOMOM1, R2
8312 045214 012703 000054 MOV #S1MOM1, R3
8313 045220 050502 BIS R5, R2
8314 045222 050503 BIS R5, R3
8315 045224 012700 045172 MOV #VSG1, R0 ;MAKE TEST-CODE HIT IN
8316 045230 012701 001000 MOV #1000, R1 ;GROUP 0
8317 045234 010337 177746 1$: MOV R3, @#CONTRL;FORCE REPLACE GROUP 1
8318 045240 005760 002000 TST 2000(R0)
8319 045244 010237 177746 MOV R2, @#CONTRL ;FORCE REPLACE GROUP 0
8320 045250 005720 TST (R0)+
8321 045252 077110 SOB R1, 1$
8322 045254 012700 114704 MOV #TSTDAT, R0 ;MAKE TEST-DATA HIT IN
8323 045260 012701 001000 MOV #1000, R1 ;GROUP 1
8324 045264 010337 177746 MOV R3, @#CONTRL ;FORCE REPLACE GROUP 1
8325 045270 042737 000014 177746 BIC #MOM1, @#CONTRL
8326 045276 005720 2$: TST (R0)+
8327 045300 077102 SOB R1, 2$
8328 045302 042737 000040 177746 BIC #S1, @#CONTRL
8329 045310 052737 000020 177746 BIS #S0, @#CONTRL ;FORCE REPLACE GROUP 0
8330 045316 052737 000400 177746 BIS #FCAC, @#CONTRL ;FLUSH CACHE
8331 045324 013704 177746 MOV @#CONTRL, R4
8332 045330 074504 XOR R5, R4 ;CHECK IF VSIU COMPLEMENTED
8333 045332 032704 020000 BIT #VSIU, R4
8334 045336 001004 BNE 3$
8335 045340 013737 177746 001556 MOV @#CONTRL, $REG0
8336 045346 104064 ERROR 64 ;VSIU DID NOT SWITCH WHEN
8337 ;CACHE-FLUSH WAS DONE
8338 045350 052737 000014 177746 3$: BIS #MOM1, @#CONTRL;MAKE TEST-CODE HIT IN
8339 045356 012700 045172 MOV #VSG1, R0 ;GROUP 0
8340 045362 012701 001000 MOV #1000, R1
8341 045366 005720 4$: TST (R0)+
8342 045370 077102 SOB R1, 4$
8343 045372 042737 000014 177746 BIC #MOM1, @#CONTRL
8344 045400 012700 114704 MOV #TSTDAT, R0 ;REFERENCE TEST-DATA AND CHECK
8345 045404 012701 000400 MOV #400, R1 ;THAT IT IS A MISS. NOTE
8346 045410 005710 5$: TST (R0) ;SETTING CACHE-FLUSH SHOULD
8347 045412 032737 000010 177752 BIT #10, @#HITMIS ;HAVE INVALIDATED GROUP
8348 045420 001411 BEQ 6$
8349 045422 013737 177746 001556 MOV @#CONTRL, $REG0
8350 045430 012737 000001 001560 MOV #1, $REG1 ;GROUP NO.
8351 045436 010037 001562 MOV R0, $REG2 ;TEST DATA ADDRESS
8352 045442 104066 ERROR 66 ;TEST-DATA WAS NOT A MISS.
8353 ;TEST DATA WAS MADE A HIT
8354 ;IN GROUP 1 AND THEN CACHE-
8355 ;FLUSH WAS DONE. CACHE-FLUSH
    
```

```
8356                                     ;SHOULD HAVE INVALIDATED GROUP
8357                                     ;1'S CACHED DATA. HENCE, THE
8358                                     ;TEST DATA REFERENCE SHOULD
8359                                     ;HAVE BEEN A MISS.
8360                                     ;PROBABLE FAILURE:
8361 045444 062700 000004      6$:  ADD    #4, R0      ;VALID STORE IS NOT BEING SWITCHED
8362 045450 077121          SOB    R1, 5$      ;TO THE OTHER WHEN CACHE-FLUSH IS
8363                                     ;SET IN THE CCR
8364 045452 032737 010000 177746 7$:  BIT    #VCIP, @#CONTRL
8365 045460 001374          BNE    7$
8366 045462 012700 020000          MOV    #VSIU, R0      ;COMPLEMENT VSIU
8367 045466 074037 177746          XOR    R0, @#CONTRL
8368 045472 032737 010000 177746 8$:  BIT    #VCIP, @#CONTRL
8369 045500 001374          BNE    8$
8370 045502 052737 000014 177746          BIS    #MOM1, @#CONTRL ;MAKE TEST-CODE HIT IN
8371 045510 012700 045172          MOV    #VSG1, R0      ;GROUP 0
8372 045514 012701 001000          MOV    #1000, R1
8373 045520 005720          9$:  TST    (R0)+
8374 045522 077102          SOB    R1, 9$
8375 045524 042737 000014 177746          BIC    #MOM1, @#CONTRL
8376                                     ;THE ORIGINAL VALID STORE (WHICH
8377                                     ;WAS INVALIDATED BY CACHE FLUSH)
8378                                     ;IS IN USE AGAIN.
8379 045532 012700 114704          MOV    #TSTDAT, R0
8380 045536 012701 000400          MOV    #400, R1      ;REFERENCE THE TEST-DATA AND
8381                                     ;CHECK IT IS A MISS
8382 045542 005710          10$: TST    (R0)
8383 045544 032737 000010 177752          BIT    #10, @#HITMIS
8384 045552 001411          BEQ    11$
8385 045554 013737 177746 001556          MOV    @#CONTRL, $REG0
8386 045562 012737 000001 001560          MOV    #1, $REG1      ;GROUP NO.
8387 045570 010037 001562          MOV    R0, $REG2      ;TEST DATA ADDRESS
8388 045574 104067          ERROR 67      ;TEST-DATA REFERENCE WAS NOT A MISS (IN
8389                                     ;GROUP 1, ORIGINAL VALID STORE). CACHE-FLUSH
8390                                     ;DONE EARLIER ON THE ORIGINAL VALID STORE
8391                                     ;SHOULD HAVE RESULTED IN INVALIDATING
8392                                     ;THE VALID STORE, THUS RESULTING IN
8393                                     ;CACHE-MISS ON TEST DATA REFERENCE.
8394                                     ;PROBABLE FAILURE: VALID STORE IN USE IS NOT
8395                                     ;BEING INVALIDATED WHEN CACHE-FLUSH IS
8396                                     ;SET
8397 045576 062700 000004      11$:  ADD    #4, R0
8398 045602 077121          SOB    R1, 10$
8399 045604 012701 020000          MOV    #VSIU, R1
8400 045610 074105          XOR    R1, R5 ;TESTED VALID STORE B (1)?
8401 045612 001402          BEQ    TST47      ;EXIT
8402 045614 000137 045174          JMP    VSG1A
```

```
8403
8404
8405 ::*****
8406 :*TEST 47 CHECK CACHE TURNS OFF WHEN FLUSH IS DONE WITH IVSS SET
8407 ;THIS TEST CHECKS THAT IF CACHE-FLUSH IS DONE (SETTING CF), WHEN IVSS
8408 ;IS SET, THE VALID STORES ARE NOT SWITCHED AND THE CACHE IS TURNED
8409 ;OFF (AND A SLOW FLUSH IS PERFORMED). THUS, ANY REFERENCE TO
8410 ;A PREVIOUSLY CACHED DATA SHOULD RESULT IN CACHE MISS.
8411 ;TEST-DATA IS MADE HIT IN GROUP 0 (BEING TESTED). TEST CODE IS
```



```

8412                                     ;MADE HIT IN GROUP 1.IVSS IS SET AND A FLUSH IS DONE. PREVIOUSLY
8413                                     ;CACHED TEST-DATA IS REFERENCED TO CHECK IT IS A MISS.
8414                                     ;THE TEST IS REPEATED FOR BOTH GROUPS AND VALID STORES.
8415                                     ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8416                                     ;*****
8417 045620 000004 TST47: SCOPE
8418
8419 045622 005002 IVFC: CLR R2 ;BIT MASK FOR VSIU
8420 045624 012700 000034 MOV #SOMOM1,R0
8421 045630 012701 000054 MOV #S1MOM1,R1
8422 045634 050200 IVFCA: BIS R2,R0
8423 045636 050201 BIS R2,R1
8424 045640 010237 177746 MOV R2,@CONTRL
8425 045644 032737 010000 177746 1$: BIT #VCIP,@CONTRL
8426 045652 001374 BNE 1$
8427
8428 045654 012703 045622 MOV #IVFC,R3 ;MAKE TEST CODE BIT IN GROUP
8429 045660 012704 001000 MOV #1000,R4 ;NOT BEING TESTED
8430 045664 010037 177746 2$: MOV R0,@CONTRL
8431 045670 005763 002000 TST 2000(R3)
8432 045674 010137 177746 MOV R1,@CONTRL
8433 045700 005723 TST (R3)+
8434 045702 077410 SOB R4,2$
8435
8436 045704 042700 000014 BIC #MOM1,R0
8437 045710 042701 000014 BIC #MOM1,R1
8438
8439 045714 012703 114704 MOV #TSTDAT,R3 ;MAKE TEST-DATA HIT IN GROUP
8440 045720 012704 001000 MOV #1000,R4 ;BEING TESTED
8441
8442 045724 010037 177746 MOV R0,@CONTRL
8443 045730 005723 3$: TST (R3)+
8444 045732 077402 SOB R4,3$
8445
8446 045734 010137 177746 MOV R1,@CONTRL ;FORCE REPLACE GROUP (NOT BEING TESTED)
8447 045740 052737 040000 177746 BIS #IVSS,@CONTRL ;SET IVSS
8448
8449 045746 012705 000004 MOV #4,R5 ;BIT MASK FOR HIT/MISS REGISTER
8450 045752 012704 000400 MOV #400,R4
8451 045756 012703 116704 MOV #TSTDAT+2000,R3
8452 045762 052737 000400 177746 BIS #FCAC,@CONTRL ;FLUSH CACHE
8453 045770 005743 4$: TST -(R3) ;REFERENCE PREVIOUSLY CACHED
8454 045772 030537 177752 BIT R5,@WHITMIS ;TEST DATA& CHECK IT IS A MISS
8455 045776 001004 BNE 6$
8456 046000 162703 000002 5$: SUB #2,R3
8457 046004 077407 SOB R4,4$
8458 046006 000417 BR 7$ ;DONE
8459
8460 046010 013737 177746 001556 6$: MOV @CONTRL,$REGO
8461 046016 005037 001560 CLR $REG1 ;GROUP NO.
8462 046022 032700 000040 BIT #S1,R0 ;WHICH GROUP?
8463 046026 001403 BEQ 12$
8464 046030 012737 000001 001560 MOV #1,$REG1 ;GROUP NO
8465 046036 010337 001562 12$: MOV R3,$REG2 ;TEST DATA ADDRESS
8466 046042 104072 ERROR 7$ ;TEST DATA REFERENCE DID NOT
8467 ;REGISTER A MISS. TEST-DATA WAS
    
```

```

8468 046044 000755 BR 5$ ;MADE BIT IN A GROUP. CACHE-FLUSH
8469 ;WAS DONE, WITH IVSS SET. REFERENCE
8470 ;TO THE PREVIOUSLY CACHED DATA
8471 ;SHOULD HAVE BEEN A MISS.
8472 ;PROBABLE FAILURE: CACHE DOES NOT
8473 ;TURN OFF WHEN IVSS IS SET AND
8474 ;FLUSH IS PERFORMED
8475 ;CHECK THAT THE CACHE HAS TURNED ON AGAIN,CHECK
8476 ;THAT HITS CAN BE OBTAINED
8477
8478 046046 012703 045622 7$: MOV #IVFC,R3 ;MAKE THE TEST-CODE HIT IN GROUP NOT
8479 046052 012704 001000 MOV #1000,R4 ;BEING TESTED
8480 046056 052700 000014 BIS #MOM1,R0
8481 046062 052701 000014 BIS #MOM1,R1
8482 046066 010037 177746 8$: MOV R0,@#CONTRL
8483 046072 005763 002000 TST 2000(R3)
8484
8485 046076 010137 177746 MOV R1,@#CONTRL
8486 046102 005723 TST (R3)+
8487 046104 077410 SOB R4,8$
8488
8489 046106 042700 000014 BIC #MOM1,R0
8490 046112 042701 000014 BIC #MOM1,R1
8491
8492 046116 012703 114704 MOV #TSTDAT,R3 ;MAKE TEST-DATA HIT IN GROUP
8493 046122 012704 001000 MOV #1000,R4 ;BEING TESTED
8494
8495 046126 010037 177746 MOV R0,@#CONTRL
8496 046132 005723 9$: TST (R3)+
8497 046134 077402 SOB R4,9$
8498
8499 ;FORCE REPLACE GROUP NOT BEING
8500 046136 010137 177746 MOV R1,@#CONTRL ;TESTED
8501 046142 012703 114704 MOV #TSTDAT,R3 ;REFERENCE TEST-DATA (IN THE
8502 046146 012704 001000 MOV #1000,R4 ;GROUP BEING CHECKED) AND
8503 046152 005713 10$: TST (R3) ;MAKE SURE IT IS A HIT
8504 046154 032737 000010 177752 BIT #10,@#HITMIS ;HIT?
8505 046162 001016 BNE 11$ ;YES
8506 046164 013737 177746 001556 MOV @#CONTRL,$REG0
8507 046172 005037 001560 CLR $REG1 ;GROUP NO.
8508 046176 032700 000040 BIT #S1,R0 ;WHICH GROUP?
8509 046202 001403 BEQ 13$
8510 046204 012737 000001 001560 MOV #1,$REG1 ;GROUP NO
8511 046212 010337 001562 13$: MOV R3,$REG2 ;TEST DATA ADDRESS
8512 046216 104073 ERROR 73 ;PREVIOUSLY CACHED TEST-DATA
8513 ;WAS REFERENCED BUT IT
8514 ;WAS NOT A HIT.
8515 ;POSSIBLE FAULT: CACHE DID NOT
8516 ;TURN ON AFTER HAVING TURNED
8517 ;OFF (WHEN A CACHE FLUSH
8518 ;WAS DONE WITH IVSS SET).
8519
8520 046220 062703 000002 11$: ADD #2,R3
8521 046224 077426 SOB R4,10$ ;DONE?
8522 046226 052700 000014 BIS #MOM1,R0
8523 046232 052701 000014 BIS #MOM1,R1

```

8524	046236	012704	020000		MOV	#VSIU,R4	
8525	046242	074402			XOR	R4,R2	;DONE VALID STORE B?
8526	046244	001402			BEQ	14\$	
8527	046246	000137	045634		JMP	IVFCA	
8528	046252	032700	000040	14\$:	BIT	#S1,R0	;YES, DONE GROUP 1?
8529	046256	001007			BNE	TST50	::YES,EXIT

8530 046260 012700 000054
8531 046264 012701 000034
8532 046270 005002
8533 046272 000137 045634

MOV #S1MOM1,R0 ;CCR MASKS FOR GROUP 1 TESTING
MOV #SOMOM1,R1
CLR R2 ;BIT MASK FOR VALID STORE
JMP IVFCA ;GO TEST GROUP 1

8534
8535
8536
8537
8538
8539
8540
8541
8542
8543
8544

: *TEST 50 CHECK CACHE TURNS OFF ON A BACK-TO-BACK FLUSH
: 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.
: THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM

8545 046276 000004
8546 046300 005037 177746
8547 046304 032737 010000 177746
8548 046312 001374
8549 046314 012701 177774
8550 046320 012700 000007
8551 046324 012737 000400 177746
8552 046332 012737 000400 177746
8553
8554 046340 005201
8555 046342 001410
8556 046344 030037 177752
8557 046350 001773
8558 046352 013702 177752
8559 046356 010237 001556
8560 046362 104115

TST50: SCOPE
CLR @#CONTRL
1\$: BIT #VCIP,@#CONTRL
BNE 1\$
MOV #-4,R1
MOV #7,R0
MOV #FCAC,@#CONTRL ;FLUSH CACHE
MOV #FCAC,@#CONTRL ;AGAIN FLUSH THE CACHE. SINCE
;PREVIOUS FLUSH IS STILL IN
2\$: INC R1 ;PROGRESS CACHE SHOULD BE
BEQ 3\$;TURNED OFF
BIT R0,@#HITMIS ;CHECK THAT THE LAST THREE REFERENCES
BEQ 2\$;WERE MISSES
MOV @#HITMIS,R2
MOV R2,\$REGO
8561 ERROR 11\$;CACHE DID NOT TURN OFF ON
8562 ;PERFORMING A BACK-TO-BACK
8563 ;FLUSH. FOR THE PERIOD OF TIME
8564 ;THAT THE FLUSH IS BEING DONE
8565 ;AND THE CACHE IS OFF, ALL
8566 ;REFERENCES SHOULD BE FORCED
8567 046364 3\$: ;TO MAIN MEMORY (MISSES).
;EXIT

8561
8562
8563
8564
8565
8566
8567
8568
8569
8570
8571
8572
8573
8574
8575
8576
8577
8578
8579
8580
8581
8582
8583
8584
8585

: *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 INVAL-
: IDATED 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

```

8586                                     ;OTHER VALID STORE AND THEN WITH TEST-DATA IN
8587                                     ;GROUP 1.
8588                                     ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR FM
8589                                     ;*****
8590 046364 000004 TST51: SCOPE
8591 046366 005002 CBP: CLR R2 ;BIT MASK FOR VSIU
8592 046370 012700 000034 MOV #SOMOM1,R0
8593 046374 012701 000054 MOV #S1MOM1,R1
8594 046400 050200 CBPA: BIS R2,R0
8595 046402 050201 BIS R2,R1
8596 046404 010237 177746 MOV R2,@#CONTRL
8597 046410 032737 010000 177746 1$: BIT #VCIP,@#CONTRL ;SELECT VSIU
8598 046416 001374 BNE 1$
8599
8600 046420 012703 046366 MOV #CBP,R3 ;MAKE TEST-CODE HIT IN THE
8601 046424 012704 001000 MOV #1000,R4 ;GROUP NOT BEING TESTED
8602 046430 010037 177746 2$: MOV R0,@#CONTRL
8603 046434 005763 002000 TST 2000(R3)
8604 046440 010137 177746 MOV R1,@#CONTRL
8605 046444 005723 TST (R3)+
8606 046446 077410 SOB R4,2$
8607
8608 046450 042700 000014 BIC #MOM1,R0
8609 046454 042701 000014 BIC #MOM1,R1
8610
8611 046460 012703 114704 MOV #TSTDAT,R3 ;MAKE TEST-DATA HIT IN THE
8612 046464 012704 001000 MOV #1000,R4 ;GROUP BEING TESTED
8613 046470 010037 177746 MOV R0,@#CONTRL
8614 046474 005723 3$: TST (R3)+
8615 046476 077402 SOB R4,3$
8616 046500 010137 177746 MOV R1,@#CONTRL ;FORCE REPLACE IN THE GROUP NOT
8617 ;BEING TESTED
8618 046504 052737 001000 177746 BIS #UCB,@#CONTRL ;UNCONDITIONED CACHE BY-PASS
8619
8620 046512 012703 114704 MOV #TSTDAT,R3
8621 046516 012704 001000 MOV #1000,R4 ;REFERENCE THE CACHED-TEST-DATA
8622 046522 005713 4$: TST (R3) ;THE GROUP BEING TESTED
8623 046524 032737 000010 177752 BIT #10,@#HITMIS ;MISS?
8624 046532 001416 BEQ 5$ ;YES
8625
8626 046534 013737 177746 001556 MOV @#CONTRL,$REG0
8627 046542 005037 001560 CLR $REG1 ;GROUP NO.
8628 046546 032700 000040 BIT #S1,R0 ;WHICH GROUP?
8629 046552 001403 BEQ 8$
8630 046554 012737 000001 001560 MOV #1,$REG1 ;GROUP NO
8631 046562 010337 001562 8$: MOV R3,$REG2 ;TEST DATA ADDRESS
8632 046566 104074 ERROR 74 ;TEST-DATA-REFERENCE WAS NOT
8633 ;A MISS. TEST-DATA WAS PREVIOUSLY
8634 ;CACHED IN THE GROUP BEING
8635 ;TESTED. THEN IT WAS REFERENCED
8636 ;WITH CACHE BY-PASS SET. IT
8637 ;SHOULD HAVE BEEN A MISS.
8638 ;PROBABLE FAILURE : A MISS IS
8639 ;NOT BEING FORCED WHEN CACHE
8640 ;BYPASS IS SET
8641
    
```

```

8642 046570 062703 000002 5$: ADD #2,R3
8643 046574 077426 SOB R4,4$ ;DONE?
8644
8645 046576 042737 001000 177746 BIC #UCB,@#CONTRL ;CLEAR CACHE BYPASS
8646 046604 012703 114704 MOV #TSTDAT,R3 ;REFERENCE THE TEST-DATA AGAIN
8647 046610 012704 000400 MOV #400,R4 ;IT SHOULD BE A MISS
8648 046614 005713 6$: TST (R3)
8649 046616 032737 000010 177752 BIT #10,@#HITMIS ;MISS?
8650 046624 001416 BEQ 7$ ;YES
8651
8652 046626 013737 177746 001556 MOV @#CONTRL,$REGO
8653 046634 005037 001560 CLR $REG1 ;GROUP NO.
8654 046640 032700 000040 BIT #S1,R0 ;WHICH GROUP?
8655 046644 001403 BEQ 9$
8656 046646 012737 000001 001560 MOV #1,$REG1 ;GROUP NO
8657 046654 010337 001562 9$: MOV R3,$REG2 ;TEST DATA ADDRESS
8658 046660 104075 ERROR 7$ ;TEST-DATA-REFERENCE WAS NOT
8659 ;A MISS. TEST-DATA WAS PREVIOUSLY
8660 ;CACHED IN THE GROUP BEING
8661 ;TESTED. THEN IT WAS INVALIDATED
8662 ;BY REFERENCING IT WHILE
8663 ;CACHE-BYPASS WAS SET. THEN
8664 ;CACHE-BYPASS WAS CLEARED AND
8665 ;THE TEST DATA WAS REFERENCED
8666 ;AGAIN TO MAKE SURE IT WAS
8667 ;INVALIDATED.
8668 046662 062703 000004 7$: ADD #4,R3 ;PROBABLE FAILURE - CACHE-BYPASSS
8669 046666 077426 SOB R4,6$ ;DOES NOT INVALIDATE DATE
8670 ;THAT IS A HIT INSIDE THE
8671 ;CACHE
8671 046670 052700 000014 BIS #MOM1,R0
8672 046674 052701 000014 BIS #MOM1,R1
8673 046700 012704 020000 MOV #VSIU,R4
8674 046704 074402 XOR R4,R2 ;DONE BOTH VALID STORES?
8675 046706 001234 BNE CBPA ;NO
8676 046710 032700 000040 BIT #S1,R0 ;TESTED GROUP 1
8677 046714 001005 BNE TST52 ;EXIT
8678
8679 046716 012700 000054 MOV #S1MOM1,R0 ;SET UP FOR TESTING GROUP 1
8680 046722 012701 000034 MOV #SOMOM1,R1
8681 046726 000624 BR CBPA
8682
8683
8684
8685
8686
8687
8688
8689
8690
8691
8692
8693
8694
8695
8696
8697
    
```

```

:*****
:*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 (INCLDING 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
    
```

```

8698
8699 046730 000004
8700
8701 046732 012703 046732 ASRBCB: MOV #ASRBCB,R3
8702 046736 012704 001000 MOV #1000,R4 ;MAKE TEST-CODE HIT IN GROUP
8703 046742 012737 000034 177746 1$: MOV #SOMOM1,@#CONTRL;1
8704 046750 005763 002000 TST 2000(R3)
8705 046754 012737 000054 177746 MOV #S1MOM1,@#CONTRL
8706 046762 005723 TST (R3)+
8707 046764 077412 SOB R4,1$
8708 046766 042737 000014 177746 BIC #MOM1,@#CONTRL
8709 ;EXECUTE AN ASRB AND REFERENCE
8710 046774 106237 047076 ASRB @#ASLOC ;THE TEST LOCATION
8711 047000 032737 000010 177752 BIT #10,@#HITMIS
8712 047006 001412 BEQ 2$
8713
8714 047010 013737 177746 001556 MOV @#CONTRL,$REG0
8715 047016 012737 000001 001560 MOV #1,$REG1 ;GROUP NO.
8716 047024 012737 047076 001562 MOV #ASLOC,$REG2 ;TEST DATA ADDRESS
8717 047032 104076 ERROR 76 ;PREVIOUSLY CACHED TEST-LOCATION
8718 ;WHEN REFERENCED USING AN
8719 ;ASRB INSTRUCTION WAS NOT
8720 ;A MISS.
8721 ;PROBABLE FAILURE = ASRB DOES
8722 ;NOT FORCE OPERAND-REFERENCE
8723 ;TO THE MAIN MEMORY
8724
8725 047034 005737 047076 2$: TST @#ASLOC ;REFERENCE THE TEST-LOCATION
8726 047040 032737 000010 177752 BIT #10,@#HITMIS ;MISS?
8727 047046 001414 BEQ TST53 ;EXIT
8728
8729 047050 013737 177746 001556 MOV @#CONTRL,$REG0
8730 047056 012737 000001 001560 MOV #1,$REG1 ;GROUP NO.
8731 047064 012737 047076 001562 MOV #ASLOC,$REG2 ;TEST DATA ADDRESS
8732 047072 104077 ERROR 77 ;BYTE-OPERAND (OF ASRB) WAS
8733 ;NOT INVALIDATED WHEN ASRB
8734 ;WAS EXECUTED ON A CACHED
8735 ;LOCATION
8736 047074 000401 BR TST53 ;EXIT
8737
8738 047076 000000 ASLOC: .WORD 0
8739
8740
8741
8742
8743 ;*****
8744 ;*TEST 53 CHECK CACHE VALID STORE PARITY CHECKER
8745 ;THIS TEST FORCES VALID STORE PARITY ERROR IN THE FOUR
8746 ;VALID STORES AND CHECKS THE PARITY CHECKERS.
8747 ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8748 ;*****
8748 047100 000004 TST53: SCOPE
8749 047102 013700 MOV @#MSER,R0
8750 047106 010037 177744 MOV R0,@#MSER
8751 047112 005002 CVSPE: CLR R2 ;BIT MASK FOR VSIU
8752 047114 012704 000034 MOV #SOMOM1,R4 ;SET UP BIT MASKS TO CHECK
8753 047120 012705 000054 MOV #S1MOM1,R5 ;GROUP 1 FIRST
    
```

```

8754 047124 050204          CVSPEA: BIS      R2,R4
8755 047126 050205          BJS      R2,R5
8756 047130 010237 177746          MOV     R2,@#CONTRL
8757 047134 032737 010000 177746 1$: BIT    #VCIP,@#CONTRL
8758 047142 001374          BNE     1$
8759
8760 047144 010437 177746          MOV     R4,@#CONTRL
8761 047150 005737 051224          TST    @#2$+2000
8762 047154 010537 177746          MOV     R5,@#CONTRL ;MAKE 'NOP' HIT IN GROUP BEING
8763 047160 005737 047224          TST    @#2$ ;TESTED
8764
8765 047164 032704 000020          BIT    #S0,R4 ;TESTING GROUP 1?
8766 047170 001004          BNE     13$ ;YES
8767 047172 042737 000004 177746          BIC    #M0,@#CONTRL ;TESTING GROUP 0,FORCE MISS GROUP 1
8768 047200 000403          BR     14$
8769 047202 042737 000010 177746 13$: BIC    #M1,@#CONTRL ;TESTING GROUP 1,FORCE MISS GROUP 0
8770
8771 047210 012737 047246 000114 14$: MOV    #3$,@#114 ;SETUP PARITY ERROR TRAP VECTOR
8772
8773 047216 052737 002000 177746          BIS    #FVPE,@#CONTRL ;FORCE VALID STORE PARITY ERROR
8774
8775 047224 000240          2$: NOP ;REFERENCE OF THIS INSTRUCTION
8776 ;WILL FORCE A VALID STORE
8777 ;PARITY ERROR. TRAP TO 114
8778 ;SHOULD OCCUR.
8779 047226 042737 002000 77746          BIC    #FVPE,@#CONTRL ;USER FVPE IF STILL SET
8780 047234 013737 177746 001556          MOV    @#CONTRL,$REGO
8781 047242 104103          ERROR  103 ;VALID STORE PARITY ERROR WAS
8782 ;FORCED BY SETTINGS FVPE IN
8783 ;CCR AND MAKING A REFERENCE
8784 ;TO A LOC THAT WAS MADE
8785 ;A HIT. PARITY ERROR TRAP
8786 ;DID NOT OCCUR ON DETECTING
8787 ;THAT PARITY ERROR IN VALID
8788 047244 000411          BR     6$ ;STORE
8789
8790 047246          3$: ;ENTER HERE IF A PARITY ERROR
8791 ;TRAP OCCURS AS EXPECTED
8792 047246 022626          CMP    (SP)+,(SP)+ ;POP THE STACK
8793 047250 032737 002000 177746 5$: BIT    #FVPE,@#CONTRL ;FVPE CLEARED AFTER PARITY ERROR?
8794 047256 001404          BEQ    6$
8795 047260 013737 177746 001556          MOV    @#CONTRL,$REGO
8796 047266 104103          ERROR  105 ;FVPE DID NOT GET CLEARED
8797 ;AFTER VALID STORE PARITY ERROR
8798 ;OCCURED
8799
8800 047270 032737 100000 177746 6$: BIT    #VSPE,@#CONTRL ;DID-VALID-STORE-PARITY-ERROR SET?
8801 047276 001004          BNE     7$
8802 047300 013737 177746 001556          MOV    @#CONTRL,$REGO
8803 047306 104106          ERROR  106 ;VALID-STORE-PARITY-ERROR BIT
8804 ;DID NOT SET IN CCR WHEN
8805 ;PARITY ERROR FROM V-STORE)
8806 047310 005003          7$: CLR    R3 ;WAS FORCED
8807 ;TESTING GROUP 1?
8808 047312 032705 000040          BIT    #S1,R5
8809 047316 001003          BNE     8$ ;YES
    
```



```

8810 047320 012700 000020      MOV    #BIT4,R0      ;SET BIT MASK FOR GROUP 0
8811 047324 000402              BR     9$
8812 047326 012700 000040      8$: MOV    #BIT5,R0      ;SET BIT MASK FOR GROUP 1
8813 047332 013701 177744      9$: MOV    @MMSER,R1     ;GROUP IN WHICH THE PARITY ERROR
8814 047336 042701 177716      BIC    #^CBIT4+^CBIT5,R1 ;OCCURED) WAS SET IN MEMORY
8815 047342 020100              CMP    R1,R0         ;SYSTEM ERROR REGISTER
8816 047344 001413              BEQ    10$
8817 047346 013737 177746 001556  MOV    @CONTRL,$REG0
8818 047354 013737 177744 001560  MOV    @MMSER,$REG1
8819 047362 010337 001562      MOV    R3,$REG2     ;GROUP NO. BEING TESTED
8820 047366 010037 001564      MOV    R0,$REG3     ;EXPECTED BITS (4,5) IN MEM SY
8821                          ;ERROR REGISTER
8822 047372 104107      ERROR  107         ;PROPER BITS (ADDRESS MEMORY)
8823                          ;PARITY ERROR.(4,5) WERE NOT
8824                          ;SET IN MEM-SYS.ERROR
8825                          ;REGISTER WHEN VALID STORE
8826                          ;PARITY ERROR WAS FORCED.
8827
8828 047374 013703 177746      10$: MOV    @CONTRL,R3   ;DID VSIU BIT SWITCH AFTER
8829 047400 074203              XOR    R2,R3        ;PARITY ERROR TRAP?
8830 047402 032703 020000      BIT    #VSIU,R3
8831 047406 001404              BEQ    11$          ;NO. OK
8832 047410 013737 177746 001556  MOV    @CONTRL,$REG0
8833 047416 104110      ERROR  110         ;VSIU SWITCHED WHEN A VALID
8834                          ;STORE PARITY ERROR WAS
8835                          ;FORCED. IT SHOULD NOT.
8836 047420 013701 177744      11$: MOV    @MMSER,R1   ;CLEAR MEMORY SYSTEM ERROR
8837 047424 010137 177744      MOV    R1,@MMSER   ;REGISTER
8838 047430 005737 177744      TST    @MMSER      ;CLEARED?
8839 047434 001404              BEQ    12$
8840 047436 013737 177744 001556  MOV    @MMSER,$REG0
8841 047444 104111      ERROR  111         ;MEMORY SYSTEM ERROR REGISTER
8842                          ;WOULD NOT BE CLEARED BY
8843                          ;WRITING ITS CONTENTS BACK
8844                          ;INTO ITSELF. NOTE PREVIOUSLY
8845                          ;A VALID STORE PARITY ERROR
8846                          ;OCCURED THIS SETTINGS ?
8847                          ;BIT 4 OR 5 IN IT.
8848 047446 012700 020000      12$: MOV    #VSIU,R0
8849 047452 074002              XOR    R0,R2        ;DONE VALID STORE B?
8850 047454 001223              BNE    CVSPEA       ;GO CHECK VALID STORE B
8851
8852 047456 032705 000020      BIT    #S0,R5       ;CHECKED GROUP 0?
8853 047462 001007              BNE    TST54        ;EXIT
8854
8855 047464 012704 000054      MOV    #S1MOM1,R4   ;SET UP BIT MASKS TO CHECK
8856 047470 012705 000034      MOV    #S0MOM1,R5   ;GROUP 0
8857 047474 005002              CLR    R2           ;BUT MASK FOR VALID STORE
8858 047476 000137 047124      JMP    CVSPEA
8859
8860      ;*****
8861      ;*TEST 54      CHECK THAT CACHE-MISS OCCURS ON A VALID STORE PARITY ERROR
8862      ;THIS TEST FORCES A VALID STORE PARITY ERROR AND CHECKS
8863      ;THAT A MISS OCCURS ON THE REFERENCE THAT CAUSED
8864      ;THE PARITY ERROR. THE CACHE LOCATION THAT GAVE THE
8865      ;PARITY ERROR IS INVALIDATED AND A SLOW CYCLE IS
    
```

```

8866                                     ;PERFORMED WITH THE 'DISABLE TRAPS' BIT OF THE
8867                                     ;CACHE CONTROL REGISTER SET, THUS A PARITY ERROR
8868                                     ;TRAP WILL NOT OCCUR. THIS IS DONE SO THAT THE
8869                                     ;HIT-MISS REGISTER CAN BE READ WITHOUT LOSING
8870                                     ;THE INFORMATION CONTAINED IN IT.
8871                                     ;THIS TEST WILL ONLY BE EXECUTED ON A KB-11CM,E, OR EM
8872                                     ;*****
8873 047502 000004 TST54: SCOPE
8874 047504 005002 VSCM: CLR R2 ;BIT MASK FOR VSIU
8875                                     ;TRAPS
8876 047506 012704 000034 MOV #SOMOM1,R4 ;SET BIT MASKS TO CHECK
8877 047512 012705 000054 MOV #S1MOM1,R5 ;GROUP 1 FIRST
8878 047516 050204 VSCMA: BIS R2,R4
8879 047520 050205 BIS R2,R5
8880 047522 010237 177746 MOV R2,@#CONTRL
8881 047526 032737 010000 177746 1$: BIT #VCIP,@#CONTRL
8882 047534 001374 BNE 1$
8883
8884 047536 010437 177746 MOV R4,@#CONTRL ;MAKE 'NOP' LIST IN GROUP
8885 047542 005737 051572 TST @#2$+2000 ;BEING TESTED
8886 047546 010537 177746 MOV R5,@#CONTRL
8887 047552 005737 047572 TST @#2$
8888 047556 042737 000014 177746 BIC #MOM1,@#CONTRL
8889 047564 052737 002001 177746 BIS #FVPE+DT,@#CONTRL ;FORCE VALID STORE PARITY ERROR
8890
8891 047572 000240 2$: NOP ;REFERENCE OF THIS INSTRUCTION
8892                                     ;WILL FORCE A VALID STORE
8893                                     ;PARITY ERROR
8894
8895 047574 032737 000010 177752 BIT #10,@#HITMIS ;CHECK THAT THE REFERENCE
8896 047602 001407 BEQ 3$ ;WHICH CAUSED THE V-STORE
8897 047604 013737 177746 001556 MOV @#CONTRL,$REGO ;PARITY ERROR WAS A MISS
8898 047612 013737 177744 001560 MOV @#MSER,$REG1 ;TEXT-DATA REFERENCE WHICH
8899 047620 104104 ERROR 104 ;CAUSED A PARITY ERROR
8900                                     ;(IN THE VALID STORE)
8901                                     ;SHOULD HAVE BEEN A MISS-
8902                                     ;IT WAS NOT.
8903
8904 047622 052737 100000 177746 3$: BIT #VSPE,@#CONTRL ;DID VALID STORE PARITY ERROR
8905                                     ;SET?
8906 047630 001004 BNE 4$
8907 047632 013737 177746 001556 MOV @#CONTRL,$REGO
8908 047640 104106 ERROR 106 ;VALID STORE PARITY ERROR BIT
8909                                     ;DID NOT SET IN CCR WHEN
8910                                     ;PARITY ERROR FROM (V-STORE)
8911                                     ;WAS FORCFD
8912
8913 047642 052737 100000 177746 4$: BIS #VSPE,@#CONTRL ;CLEAR VSPE
8914 047650 032737 100000 177746 BIT #VSPE,@#CONTRL ;CHECK
8915 047656 001404 BEQ 5$
8916 047660 013737 177746 001556 MOV @#CONTRL,$REGO
8917 047666 104112 ERROR 112 ;VALID STORE PARITY ERROR
8918                                     ;BIT COULD NOT BE CLEARED IN CCR
8919
8920 047670 013737 177744 177744 5$: MOV @#MSER,@#MSER ;CLEAR MEMORY SYSTEM ERROR
8921                                     ;REGISTER
    
```

```

8922 047676 012700 020000      MOV      #VSIU,R0
8923 047702 074002              XOR      R0,R2          ;DONE VALID STORE B?
8924 047704 001304              BNE     VSCMA          ;GO CHECK V-STORE B
8925 047706 032705 000020      BIT     #S0,R5        ;CHECKED GROUP 0
8926 047712 001007              BNE     6$
8927
8928 047714 012704 000054      MOV     #S1MOM1,R4     ;SET UP BIT MASKS ITS CHECK
8929 047720 012705 000034      MOV     #S0MOM1,R5    ;GROUP 0
8930 047724 005002              CLR     R2
8931 047726 000137 047516      JMP     VSCMA
8932
8933 047732 005037 177746      6$:    CLR     @#CONTRL
8934 047736
8935
8936
8937
8938
8939
8940
8941 047736 000004
8942 047740 012737 000012 001676  TST55:  SCOPE
8943 047746 105737 001714      MOV     #12,$TIMES    ;;DO 12 ITERATIONS
8944 047752 001003              TSTB   KB11CM
8945 047754 005737 001712      BNE     5$            ;BR IF MOIFIED 11/70 (KB11CM)
8946 047760 001444              TST    KB11E          ;IS IT A KB11-E OR KB11-EM?
8947 047762 012700 172300      BEQ    TST56          ;
8948 047766 005010 4$:    MOV     #KIPDR0,R0    ;POINT TO KIPDR0
8949 047770 032710 100000      CLR    (R0)          ;CLEAR KIPDR
8950 047774 001405              BIT    #BYP,(R0)     ;DID BYP CLEAR?
8951 047776 010037 001556      BEQ    1$            ;BRANCH IF YES
8952 050002 011037 001560      MOV    R0,$REG0
8953 050006 104123              MOV    (R0),$REG1
8954
8955 050010 052710 100000      ERROR  123          ;BYP STUCK SET
8956 050014 032710 100000      1$:    BIS    #BYP,(R0)   ;SET BYP
8957 050020 001005              BIT    #BYP,(R0)    ;IS IT SET?
8958 050022 010037 001556      BNE     2$            ;BRANCH IF YES
8959 050026 011037 001560      MOV    RC,$REG0
8960 050032 104124              MOV    (R0),$REG1
8961
8962 050034 042710 100000      ERROR  124          ;BYP STUCK CLEAR
8963 050040 032710 100000      2$:    BIC    #BYP,(R0) ;CLEAR BYP
8964 050044 001405              BIT    #BYP,(R0)    ;IS IT CLEAR?
8965 050046 010037 001556      BEQ    3$            ;BRANCH IF YES
8966 050052 011037 001560      MOV    R0,$REG0
8967 050056 104123              MOV    (R0),$REG1
8968
8969 050060 062700 000002      ERROR  123          ;BYP STUCK SET
8970 050064 020027 172340      3$:    ADD    #2,R0       ;POINT TO NEXT PDR
8971 050070 001336              CMP    R0,#KDPDR7+2 ;ARE WE FINISHED?
8972
8973
8974
8975
8976
8977 050072 000004              BNE     4$            ;BRANCH IF NOT

```

```

;*****
;*TEST 56      CHECK BYP ON SUPERVISOR PAGE BITS
;*****
;THIS TEST IS EXECUTED ONLY ON KB11-E, KB11-EM, AND MODIFIED KB11-B/C (KB11CM).
;*****
TST56:  SCOPE

```

```

8978 050074 012737 000012 001676      MOV      #12,$TIMES      ;;DO 12 ITERATIONS
8979 050102 105737 001714                TSTB     KB11CM
8980 050106 001003                BNE      5$             ;BR IF MOIFIED 11/70 (KB11CM)
8981 050110 005737 001712                TST      KB11E          ;IS IT A KB11-E OR KB11-EM?
8982 050114 001444                BEQ      TST57          ;
8983 050116 012700 172200      5$:     MOV      #SIPDR0,R0    ;POINT TO SIPDR0
8984 050122 005010      4$:     CLR      (R0)          ;CLEAR SIPDR
8985 050124 032710 100000                BIT      #BYP,(R0)      ;DID BYP CLEAR?
8986 050130 001405                BEQ      1$             ;BRANCH IF YES
8987 050132 010037 001556                MOV      R0,$REG0
8988 050136 011037 001560                MOV      (R0),$REG1
8989 050142 104130                ERROR    130            ;BYP STUCK SET
8990
8991 050144 052710 100000      1$:     BIS      #BYP,(R0)    ;SET BYP
8992 050150 032710 100000                BIT      #BYP,(R0)      ;DID IT SET?
8993 050154 001005                BNE      2$             ;BRANCH IF YES
8994 050156 010037 001556                MOV      R0,$REG0
8995 050162 011037 001560                MOV      (R0),$REG1
8996 050166 104131                ERROR    131            ;BYP STUCK CLEAR
8997
8998 050170 042710 100000      2$:     BIC      #BYP,(R0)    ;CLEAR BYP
8999 050174 032710 100000                BIT      #BYP,(R0)      ;IS IT CLEAR?
9000 050200 001405                BEQ      3$             ;BRANCH IF YES
9001 050202 010037 001556                MOV      R0,$REG0
9002 050206 011037 001560                MOV      (R0),$REG1
9003 050212 104130                ERROR    130            ;BYP STUCK SET
9004
9005 050214 062700 000002      5$:     ADD      #2,R0        ;POINT TO NEXT PDR
9006 050220 020027 172240                CMP      R0,#SDPDR7+2   ;ARE WE FINISHED?
9007 050224 001336                BNE      4$             ;BRANCH IF NO
9008
9009
9010
9011
9012 050226 000004                TST57: SCOPE
9013 050230 012737 000012 001676      MOV      #12,$TIMES      ;;DO 12 ITERATIONS
9014 050236 105737 001714                TSTB     KB11CM
9015 050242 001003                BNE      5$             ;BR IF MOIFIED 11/70 (KB11CM)
9016 050244 005737 001712                TST      KB11E          ;IS IT A KB11-E OR KB11-EM?
9017 050250 001444                BEQ      TST60          ;
9018 050252 012700 177600      5$:     MOV      #UIPDR0,R0    ;POINT TO UIPDR0
9019 050256 005010      4$:     CLR      (R0)          ;CLEAR UIPDR
9020 050260 032710 100000                BIT      #BYP,(R0)      ;DID BYP CLEAR?
9021 050264 001405                BEQ      1$             ;BRANCH IF YES
9022 050266 010037 001556                MOV      R0,$REG0
9023 050272 011037 001560                MOV      (R0),$REG1
9024 050276 104132                ERROR    132            ;BYP STUCK SET
9025
9026 050300 052710 100000      1$:     BIS      #BYP,(R0)    ;SET BYP
9027 050304 032710 100000                BIT      #BYP,(R0)      ;IS IT SET?
9028 050310 001005                BNE      2$             ;BRANCH IF YES
9029 050312 010037 001556                MOV      R0,$REG0
9030 050316 011037 001560                MOV      (R0),$REG1
9031 050322 104133                ERROR    133            ;BYP STUCK CLEAR
9032
9033 050324 042710 100000      2$:     BIC      #BYP,(R0)    ;CLEAR BYP
    
```

 ;*TEST 57 CHECK BYP ON USER PAGE BITS
 ;*THIS TEST IS EXECUTED ONLY ON KB11-E, KB11-EM, AND MODIFIED KB11-B/C (KB11CM).

```

9034 050330 032710 100000 BIT #BYP,(R0) ;IS IT CLEAR?
9035 050334 001405 BEQ 3$ ;BRANCH IF YES
9036 050336 010037 001556 MOV RO,$REG0
9037 050342 011037 001560 MOV (R0),$REG1
9038 050346 104132 ERROR 132 ;BYP STUCK SET
9039
9040 050350 062700 000002 3$: ADD #2,R0 ;POINT TO NEXT PDR
9041 050354 020027 177640 CMP RO,#UDPDR7+2 ;ARE WE FINISHED?
9042 050360 001336 BNE 4$ ;BRANCH IF NO
9043
9044 ::*****
9045 :*TEST 60 CHECK CACHE BYPASS ON VIRTUAL PAGE (KB11CM)
9046 ;*THIS TEST IS EXECUTED ONLY ON KB11-EM AND 11/74
9047 ::*****
9048 TST60: SCOPE
9049
9050 050364 013746 177776 MOV @#PS,-(SP) ;CLEAR T BIT IF SET
9051 050370 042716 000020 BIC #20,(SP)
9052 050374 012746 050402 MOV #1$,-(SP)
9053 050400 000002 RTI
9054 050402 1$:
9055
9056 050402 105737 001714 TSTB KB11CM
9057 050406 001005 BNE VPBP
9058 050410 105737 001713 TSTB KB11EM ;IS IT A KB11-EM?
9059 050414 001002 BNE VPBP ;BR IF YES
9060 050416 000137 051214 JMP VPBPE
9061 050422 012704 100000 VPBP: MOV #100000,R4 ;INITIALIZE APF, PAGE PAR 4
9062 050426 012705 172350 MOV #KIPAR4,R5
9063 050432 012703 172310 MOV #KIPDR4,R3
9064 050436 005037 177746 VPBPA: CLR @#CONTRL
9065 050442 032737 010000 177746 1$: BIT #VCIP,@#CONTRL ;WAIT FOR VCIP TO CLEAR
9066 050450 001374 BNE 1$
9067
9068 050452 012700 050422 MOV #VPBP,R0 ;MAKE TEST CODE HIT IN GROUP 1
9069 050456 012701 001000 MOV #1000,R1
9070 050462 012737 000034 177746 2$: MOV #SOMOM1,@#CONTRL
9071 050470 005760 002000 TST 2000(R0)
9072 050474 012737 000054 177746 MOV #S1MOM1,@#CONTRL
9073 050502 005720 TST (R0)+
9074 050504 077112 SOB R1,2$
9075
9076 050506 042737 000014 177746 BIC #MOM1,@#CONTRL
9077 050514 012700 062764 MOV #TSTD1,R0 ;MAKE TEST-DATA HIT IN
9078 050520 012701 001000 MOV #1000,R1 ;GROUP 0
9079 050524 012737 000020 177746 MOV #S0,@#CONTRL
9080 050532 005720 9$: TST (R0)+
9081 050534 077102 SOB R1,9$
9082
9083 050536 012737 000040 177746 MOV #S1,@#CONTRL ;FORCE REPLACE GROUP 1
9084
9085 050544 005037 172340 CLR @#KIPAR0 ;MAP 0-4K VIRTUAL INTO
9086 050550 012737 077406 172300 MOV #77406,@#KIPDR0 ;0-4K PHYSICAL (TEST PROGRAM)
9087 050556 012737 000200 172342 MOV #200,@#KIPAR1 ;MAP 4-8K VIRTUAL INTO
9088 050564 012737 077406 172302 MOV #77406,@#KIPDR1 ;4-8K PHYSICAL (TEST PROGRAM)
9089 050572 012737 000400 172344 MOV #400,@#KIPAR2 ;MAP 8-12K VIRTUAL TO

```

```

9090 050600 012737 077406 172304      MOV      #77406,@#KIPDR2 ;8-12K PHYSICAL
9091 050606 012737 000600 172346      MOV      #600,@#KIPAR3  ;MAP 12-16K VIRTUAL TO
9092 050614 012737 077406 172306      MOV      #77406,@#KIPDR3 ;12-16K PHYSICAL
9093 050622 012737 177600 172356      MOV      #177600,@#KIPAR7;MAP I/O PAGE THROUGH PAGE7
9094 050630 012737 077406 172316      MOV      #77406,@#KIPDR7
9095
9096                                     ;SET UP PAR,PDR REGISTERS TO
9097                                     ;MAP THE TEST DATA BUFFER THRU THE
9098                                     ;VIRTUAL PAGE BEING TESTED
9099 050636 012702 062764      MOV      #TSTDT1,R2      ;PHYSICAL ADDRESS
9100 050642 010200      MOV      R2,R0           ;COPY IT
9101 050644 072027 177772      ASH      #-6,R0          ;FORM THE PAF (BLOCK #)
9102 050650 010015      MOV      R0,(R5)         ;SET UP PAF INSIDE THE PAR
9103 050652 012713 010406      MOV      #10406,(R3)     ;SET UP PDR
9104
9105                                     ;FORM THE VIRTUAL ADDRESS FOR
9106                                     ;THE TEST DATA BUFFER
9107 050656 042702 177700      BIC      #177700,R2      ;CLEAR APF BIT POSITIONS
9108 050662 050402      BIS      R4,R2           ;SET APF BITS TO POINT TO THE
9109 050664 010200      MOV      R2,R0           ;PAR FOR THE VIRTUAL PAGE BEING
9110                                     ;TESTED
9111                                     ;R2 CONTAINS THE VIRTUAL ADDRESS
9112                                     ;OF THE TEST DATA BUFFER
9113 050666 012737 000020 172516      MOV      #20,@#MMR3     ;ENABLE KT - 22 BIT MODE
9114 050674 012737 000001 177572      MOV      #1,@#MMR0
9115
9116 050702 012701 001000      MOV      #1000,R1        ;COUNT
9117 050706 005712      TST      (R2)
9118 050710 032737 000010 177752      BIT      #10,@#HITMIS   ;HIT?
9119 050716 001021      BNE      4$              ;YES
9120 050720 013737 177746 001556      MOV      @#CONTRL,$REG0
9121 050726 010537 001560      MOV      R5,$REG1        ;PAR ADDRESS
9122 050732 011537 001562      MOV      (R5),$REG2      ;PAR CONTENTS
9123 050736 011337 001564      MOV      (R3),$REG3      ;PDR CONTENTS
9124 050742 010237 001566      MOV      R2,$REG4        ;TEST DATA ADDRESS (VA)
9125 050746 005037 177572      CLR      @#MMR0          ;TURN OFF MEM MAN
9126 050752 104125      ERROR    125             ;TEST-DATA-BUFFER WAS REFERENCED,
9127                                     ;IT WAS FOUND TO BE A MISS.
9128                                     ;SHOULD HAVE BEEN A HIT
9129                                     ;BECAUSE IT WAS MADE HIT
9130                                     ;IN GROUP 0 BEFORE REFERENCING
9131 050754 012737 000001 177572      MOV      #1,@#MMR0      ;TURN MM BACK ON
9132 050762 062702 000002      ADD      #2,R2
9133 050766 077131      SOB      R1,3$
9134 050770 010002      MOV      R0,R2           ;COPY VIRTUAL ADDRESS OF TEST-DATA BUFFER
9135 050772 052713 100000      BIS      #BYP,(R3)       ;SET BYPASS IN PDR
9136 050776 012701 001000      MOV      #1000,R1        ;NOW REFERENCE THE TEST LOCATIONS
9137                                     ;THAT WERE MADE HITS PREVIOUSLY
9138 051002 005710      TST      (R0)           ;CHECK THEY ARE BEING BYPASSED
9139 051004 032737 000010 177752      BIT      #10,@#HITMIS   ;MISS?
9140 051012 001421      BEQ      6$              ;YES
9141 051014 013737 177746 001556      MOV      @#CONTRL,$REG0
9142 051022 010537 001560      MOV      R5,$REG1        ;PAR ADDRESS
9143 051026 011537 001562      MOV      (R5),$REG2      ;PAR CONTENTS
9144 051032 011337 001564      MOV      (R3),$REG3      ;PDR CONTENTS
9145 051036 010037 001566      MOV      R0,$REG4        ;TEST DATA ADDRESS (VA)

```

```

9146 051042 005037 177572 CLR @MMR0 ;TURN OFF MM
9147 051046 104126 ERROR 126 ;TEST DATA WAS NOT A MISS WHEN
9148 ;IT WAS REFERENCED WITH CACHE
9149 ;BYPASS (ON VIRTUAL PAGE) SET.
9150 ;TEST DATA WAS PREVIOUSLY MADE HIT
9151 ;IN GROUP 0. IT WAS MAPPED
9152 ;THROUGH A PAR,PDR SET
9153 ;(BEING TESTED). BYPASS BIT WAS
9154 ;SET IN THE PDR AND TEST-LOC
9155 ;WAS REFERENCED. IT SHOULD HAVE
9156 ;BEEN A MISS (BECAUSE OF BYPASS)
9157 ;PROBABLE FAULT: SETTING CACHE
9158 ;BYPASS IN PDR DOES NOT BYPASS
9159 ;VIRTUAL REFERENCES MAPPED THRU
9160 ;THAT PAGE.
9161 051050 012737 000001 177572 MOV #1,@MMR0 ;TURN MM BACK ON
9162
9163
9164 051056 062700 000002 6$: ADD #2,R0
9165 051062 077131 SOB R1,5$
9166 051064 042713 100000 BIC #BYP,(R3) ;CLEAR BYPASS IN PDR
9167 051070 012701 001000 MOV #1000,R1 ;REFERENCE THE TEST-DATA AND
9168 ;MAKE SURE IT WAS INVALIDATED
9169 ;ON PREVIOUS BYPASS
9170 051074 005712 7$: TST (R2) ;REFERENCE TEST DATA
9171 051076 032737 000010 177746 BIT #10,@CONTRL ;MISS?
9172 051104 001421 BEQ 8$
9173 051106 013737 177746 001556 MOV @CONTRL,$REG0
9174 051114 010537 001560 MOV R5,$REG1 ;PAR ADDRESS
9175 051120 011537 001562 MOV (R5),$REG2 ;PAR CONTENTS
9176 051124 011337 001564 MOV (R3),$REG3 ;PDR CONTENTS
9177 051130 010237 001566 MOV R2,$REG4 ;TEST DATA ADDRESS (VIRTUAL ADDRESS)
9178 051134 005037 177572 CLR @MMR0 ;TURN OFF MM
9179 051140 104127 ERROR 127 ;TEST-DATA REFERENCE WAS NOT
9180 ;A MISS. PROBABLE FAILURE:
9181 ;PREVIOUSLY CACHED TEST DATA
9182 ;LOCATIONS WERE NOT INVALIDATED
9183 ;(IN THE CACHE) WHEN CACHE
9184 ;BYPASS WAS FORCED ON REFERENCES THROUGH
9185 ;THE VIRTUAL PAGE (BEING TESTED).
9186 051142 012737 000001 177572 MOV #1,@MMR0 ;TURN MM BACK ON
9187
9188 051150 062702 000002 8$: ADD #2,R2 ;NEXT LOCATION
9189 051154 077131 SOB R1,7$ ;DONE?
9190
9191
9192 051156 005037 177572 CLR @MMR0 ;DISABLE KT
9193 051162 005037 172516 CLR @MMR3
9194 051166 062704 020000 ADD #20000,R4 ;INITIALIZE APF FIELD MASK FOR THE
9195 051172 062705 000002 ADD #2,R5 ;NEXT PAR TO BE TESTED
9196 051176 062703 000002 ADD #2,R3 ;NEXT PDR TO BE TESTED
9197
9198 051202 020327 172316 CMP R3,#KIPDR0+16 ;DONE TESTING EVERY PDR?
9199 051206 001402 BEQ VPBPE
9200 051210 000137 050436 JMP VPBPA
9201 051214 VPBPE:

```

```

9202      ;:*****
9203
9204      .SBTTL  END OF PASS ROUTINE
9205
9206      ;*INCREMENT THE PASS NUMBER ($PASS)
9207      ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
9208      ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
9209      ;*IF THERES A MONITOR GO TO IT
9210      ;*IF THERE ,SN'T JUMP TO LOOP
9211
9212      051214      $EOP:
9213      051214      000004      SCOPE
9214      051216      005037      001502      CLR      $STSTM      ;;ZERO THE TEST NUMBER
9215      051222      005037      001676      CLR      $TIMES      ;;ZERO THE NUMBER OF ITERATIONS
9216      051226      005237      001500      INC      $PASS      ;;INCREMENT THE PASS NUMBER
9217      051232      042737      100000      001500      BIC      #100000,$PASS      ;;DON'T ALLOW A NEG. NUMBER
9218      051240      005327      DEC      (PC)+      ;;LOOP?
9219      051242      000001      $EOPCT: .WORD      1
9220      051244      003031      BGT      $DOAGN      ;;YES
9221      051246      012737      MOV      (PC)+,@(PC)+      ;;RESTORE COUNTER
9222      051250      000001      $ENDCT: .WORD      1
9223      051252      051242      $EOPCT
9224      051254      104400      051334      TYPE      , $ENDMG      ;;TYPE 'END PASS #'
9225      051260      013746      001500      MOV      $PASS,-(SP)      ;;SAVE $PASS FOR TYPEOUT
9226      051264      104410      TYPDS      ;;GO TYPE--DECIMAL ASCII WITH SIGN
9227      051266      104400      051351      TYPE      , $ENULL      ;;TYPE A NULL CHARACTER
9228      051272      013700      000042      $GET42: MOV      @#42,R0      ;;GET MONITOR ADDRESS
9229      051276      001414      BEQ      $DOAGN      ;;BRANCH IF NO MONITOR
9230      051300      012703      125252      MOV      #125252,R3
9231      051304      004737      054544      JSR      PC,CHAINQ
9232      051310      013700      000042      MOV      @#42,R0      ;;INSURE R0 CONTAINS THE MONITORS
9233      051314      001405      BEQ      $DOAGN      ;;RETURN ADDRESS
9234      051316      000005      RESET      ;;CLEAR THE WORLD
9235      051320      004710      $ENDAD: JSR      PC,(R0)      ;;GO TO MONITOR
9236      051322      000240      NOP      ;;SAVE ROOM
9237      051324      000240      NOP      ;;FOR
9238      051326      000240      NOP      ;;ACT11
9239      051330      $DOAGN:
9240      051330      000137      005144      JMP      @#LOOP      ;;RETURN
9241      051334      005015      047105      020104      $ENDMG: .ASCIZ <15><12>/END PASS #/
9242      051342      040520      051523      021440
9243      051350      000
9244      051351      377      377      000      $ENULL: .BYTE      -1,-1,0      ;;NULL CHARACTER STRING
9245
9246      ;:*****
9247
9248      .SBTTL  SCOPE HANDLER ROUTINE
9249
9250      ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
9251      ;*AND LOAD THE TEST NUMBER($STSTM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
9252      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
9253      ;*SW14=1      LOOP ON TEST
9254      ;*SW11=1      INHIBIT ITERATIONS
9255      ;*SW09=1      LOOP ON ERROR
9256      ;*SW08=1      LOOP ON TEST IN SWR<6:0>
9257      ;*CALL
    
```



```

9258      ;*      SCOPE      ;;SCOPE=IOT
9259
9260      051354      $SCOPE:
9261      051354      006137      177570      ROL      @#SWR      ;;LOOP ON PRESENT TEST?
9262      051360      100517      BMI      $OVER      ;;YES IF SW14=1
9263      ;*****START OF CODE FOR THE XOR TESTER*****
9264      051362      000416      $XSTR: BR      6$      ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
9265      ;THIS INSTRUCTION TO A 'NOP' (NOP-240)
9266      051364      013746      000004      MOV      @#ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
9267      051370      012737      051410      000004      MOV      #5$,@#ERRVEC      ;;SET FOR TIMEOUT
9268      051376      005737      177060      TS*      @#177060      ;;TIME OUT ON XOR?
9269      051402      012637      000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
9270      051406      000471      BR      $SVLAD      ;;GO TO THE NEXT TEST
9271      051410      022626      5$:      CMP      (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
9272      051412      012637      000004      MOV      (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
9273      051416      000431      BR      7$      ;;LOOP ON THE PRESENT TEST
9274      051420      6$:;*****END OF CODE FOR THE XOR TESTER*****
9275      051420      032737      000400      177570      BIT      #BIT08,@#SWR      ;;LOOP ON SPEC. TEST?
9276      051426      001412      BEQ      2$      ;;BR IF NO
9277      051430      052737      001000      177746      BIS      #BIT9,@#CONTRL      ;;TURN OFF CACHE
9278      051436      013746      177570      MOV      @#SWR,-(SP)      ;;SET DESIRED TEST NUM. FROM SWR
9279      051442      042716      000200      BIC      #5$SWR#K,(SP)      ;;STRIP AWAY UNDESIRED BITS
9280      051446      122637      001502      CMPB    (SP)+,$TSTNM      ;;ON THE RIGHT TEST?
9281      051452      001462      BEQ      $OVER      ;;BR IF YES
9282      051454      105737      001505      2$:      TSTB    $ERFLG      ;;HAS AN ERROR OCCURRED?
9283      051460      001421      BEQ      3$      ;;BR IF NO
9284      051462      123737      001517      001505      CMPB    $ERMAX,$ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
9285      051470      101015      BHI      3$      ;;BR IF NO
9286      051472      032737      001000      177570      BIT      #BIT09,@#SWR      ;;LOOP ON ERROR?
9287      051500      001404      BEQ      4$      ;;BR IF NO
9288      051502      013737      001512      001510      7$:      MOV      $LPERR,$LPADR      ;;SET LOOP ADDRESS TO LAST SCOPE
9289      051510      000443      BR      $OVER
9290      051512      105037      001505      4$:      CLRB    $ERFLG      ;;ZERO THE ERROR FLAG
9291      051516      005037      001676      CLR      $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
9292      051522      000415      BR      1$      ;;ESCAPE TO THE NEXT TEST
9293      051524      032737      004000      177570      3$:      BIT      #BIT11,@#SWR      ;;INHIBIT ITERATIONS?
9294      051532      001011      BNE      1$      ;;BR IF YES
9295      051534      005737      001500      TST      $PASS      ;;IF FIRST PASS OF PROGRAM
9296      051540      001406      BEQ      1$      ;;INHIBIT ITERATIONS
9297      051542      005237      001506      INC      $ICNT      ;;INCREMENT ITERATION COUNT
9298      051546      023737      001676      001506      CMP      $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
9299      051554      002021      BGE      $OVER      ;;RR IF MORE ITERATION REQUIRED
9300      051556      012737      000001      001506      1$:      MOV      #1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
9301      051564      013737      051634      001676      MOV      $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
9302      051572      105237      001502      $SVI AD: INCB    $TSTNM      ;;COUNT TEST NUMBERS
9303      051576      011637      001510      MOV      (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
9304      051602      011637      001512      MOV      (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
9305      051606      005037      001700      CLR      $ESCAPE      ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
9306      051612      112737      000001      001517      MOVB    #1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
9307      051620      013737      001502      177570      $OVER: MOV      $TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
9308      051626      013716      001510      MOV      $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
9309      051632      000002      RTI      ;;FIXES PS
9310      051634      000001      $MXCNT: 1      ;;MAX. NUMBER OF ITERATIONS
9311
9312      ;*****
9313

```

```

9314 .JBttl ERROR HANDLER ROUTINE
9315
9316 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
9317 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
9318 ;*AND GO TO ERTYPE ON ERROR
9319 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
9320 ;*SW15=1 HALT ON ERROR
9321 ;* HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
9322 ;*SW13=1 INHIBIT ERROR TYPEOUTS
9323 ;*SW10=1 BELL ON ERROR
9324 ;*SW09=1 LOOP ON ERROR
9325 ;*CALL
9326 ;* ERROR N ;;ERROR-EMT AND N-ERROR ITEM NUMBER
9327
9328 $ERROR:
9329 051636 105237 001505 7$: INCB $ERFLG ;;SET THE ERROR FLAG
9330 051642 001775 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO
9331 051644 013737 001502 177570 MOV $STNM,@#DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
9332 051652 005737 177570 TST @#SWR ;;HALT ON ERROR - 1?
9333 051656 100001 BPL 8$ ;;BRANCH IF NO
9334 051660 000000 HALT ;;YES--HALT
9335 051662 032737 002000 177570 8$: BIT #BIT10,@#SWR ;;BELL ON ERROR?
9336 051670 001402 BEQ 1$ ;;NO - SKIP
9337 051672 104400 001702 TYPE ,SBELL ;;RING BELL
9338 051676 005237 001514 1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS
9339 051702 011637 001520 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
9340 051706 162737 000002 001520 SUB #2,$ERRPC
9341 051714 117737 127600 001516 MOV @ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
9342 051722 032737 020000 177570 BIT #BIT13,@#SWR ;;SKIP TYPEOUT IF SET
9343 051730 001004 BNE 2$ ;;SKIP TYPEOUTS
9344 051732 004737 054744 JSR PC,ERTYPE ;;GO TO USER ERROR ROUTINE
9345 051736 104400 001707 TYPE ,SCLF
9346 051742 005737 177570 2$: TST @#SWR ;;HALT ON ERROR
9347 051746 100001 BPL 9$ ;;SKIP IF CONTINUE
9348 051750 000000 HALT ;;HALT ON ERROR.
9349 051752 022737 051320 000042 9$: CMP #SENDAD,42 ;;ACT-11?
9350 051760 001001 BNE 3$ ;;BRANCH IF NO
9351 051762 000000 HALT ;;YES
9352 051764 032737 001000 177570 3$: BIT #BIT09,@#SWR ;;LOOP ON ERROR SWITCH SET?
9353 051772 001402 BEQ 4$ ;;BR IF NO
9354 051774 013716 001512 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
9355 052000 005737 001700 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
9356 052004 001402 BEQ 5$ ;;BR IF NONE
9357 052006 013716 001700 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
9358 052012
9359 052012 012737 177777 177744 MOV #-1,@#MEMERR
9360 052020 005037 177765 CLR @#CPUERR
9361 052024 000002 RTI
    
```

```

9362
9363      ;:*****
9364
9365      .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
9366
9367      ;*SAVE R0-R5
9368      ;*CALL:
9369      ;*   SAVREG
9370      ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
9371      ;*
9372      ;*TOP---(+16)
9373      ;* +2---(+18)
9374      ;* +4---R5
9375      ;* +6---R4
9376      ;* +8---R3
9377      ;*+10---R2
9378      ;*+12---R1
9379      ;*+14---R0
9380
9381      $SAVREG:
9382      052026 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
9383      052030 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
9384      052032 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
9385      052034 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
9386      052036 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
9387      052040 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
9388      052042 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
9389      052046 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
9390      052052 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PS OF CALL
9391      052056 016646 000022      MOV      22(SP),-(SP)  ;;SAVE PC OF CALL
9392      052062 000002      RTI
9393
9394      ;*RESTORE R0-R5
9395      ;*CALL:
9396      ;*   RESREG
9397      $RESREG:
9398      052064 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PC OF CALL
9399      052070 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PS OF CALL
9400      052074 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PC OF MAIN FLOW
9401      052100 012666 000022      MOV      (SP)+,22(SP)  ;;RESTORE PS OF MAIN FLOW
9402      052104 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
9403      052106 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
9404      052110 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
9405      052112 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
9406      052114 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
9407      052116 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
9408      052120 000002      RTI
9409
9410      ;:*****
9411
9412      .SBTTL  TYPE ROUTINE
9413
9414      ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
9415      ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
9416      ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
9417      ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
    
```

```

9418      ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
9419      ;*
9420      ;*CALL:
9421      ;*1) USING A TRAP INSTRUCTION
9422      ;*      TYPE      ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
9423      ;*OR
9424      ;*      TYPE
9425      ;*      MESADR
9426      ;*
9427      ;*2) USING A JSR INSTRUCTION
9428      ;*      MOV      PS,-(SP)          ;;PUSH PROCESSOR STATUS WORD ON THE STACK
9429      ;*      JSR      PC,$TYPE          ;;CALL TYPE ROUTINE
9430      ;*      MESADDR          ;;FIRST ADDRESS OF MESSAGE
9431
9432      $TYPE:  TSTB      $TPFLG          ;;IS THERE A TERMINAL?
9433      BPL      1$          ;;BR IF YES
9434      HALT
9435      BR      3$          ;;HALT HERE IF NO TERMINAL
9436      1$:  MOV      RO,-(SP)          ;;LEAVE
9437      MOV      @2(SP),RO          ;;SAVE RO
9438      2$:  MOVB      (RO)+,-(SP)      ;;GET ADDRESS OF ASCIZ STRING
9439      BNE      4$          ;;PUSH CHARACTER TO BE TYPED ONTO STACK
9440      TST      (SP)+          ;;BR IF IT ISN'T THE TERMINATOR
9441      MOV      (SP)+,RO          ;;IF TERMINATOR POP IT OFF THE STACK
9442      3$:  ADD      #2,(SP)          ;;RESTORE RO
9443      RTI
9444      4$:  CMPB      #HT,(SP)          ;;ADJUST RETURN PC
9445      BEQ      8$          ;;RETURN
9446      CMPB      #CRLF,(SP)          ;;BRANCH IF <HT>
9447      BNE      5$          ;;BRANCH IF NOT
9448      TST      (SP)+          ;;POP <CR><LF> EQUIV
9449      TYPE      ,SCLRF
9450      BR      2$          ;;GET NEXT CHARACTER
9451      5$:  JSR      PC,$TYPEC          ;;GO TYPE THIS CHARACTER
9452      6$:  CMPB      $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
9453      BNE      2$          ;;IF NO GO GET NEXT CHAR.
9454      MOV      $NULL,-(SP)          ;;GET # OF FILLER CHARS. NEEDED
9455      ;;AND THE NULL CHAR.
9456      7$:  DECB      1(SP)          ;;DOES A NULL NEED TO BE TYPED?
9457      BLT      6$          ;;BR IF NO--GO POP THE NULL OFF OF STACK
9458      JSR      PC,$TYPEC          ;;GO TYPE A NULL
9459      DECB      $CHARCNT          ;;DON'T COUNT THE NULL AS A CHARACTER
9460      BR      7$          ;;LOOP
9461
9462      ;; RIZONTAL TAB PROCESOR
9463
9464      8$:  MOVB      #' ,(SP)          ;;REPLACE TAB WITH SPACE
9465      9$:  JSR      PC,$TYPEC          ;;TYPE A SPACE
9466      BITB      #7,$CHARCNT          ;;BRANCH IF NOT AT
9467      BNE      9$          ;;TAB STOP
9468      TST      (SP)+          ;;POP SPACE OFF STACK
9469      BR      2$          ;;GET NEXT CHARACTER
9470      $TYPEC: TSTB      @STPS          ;;WAIT UNTIL PRINTER IS READY
9471      BPI      $TYPEC
9472      MOVB      2(SP),@STPB          ;;LOAD CHAR TO BE TYPED INTO DATA REG.
9473      CMPB      #CR,2(SP)          ;;BRANCH IF
    
```

```

9474 052310 001003          BNE      1$          ;;NOT <CR>
9475 052312 105037 052332  CLR      $CHARCNT    ;;
9476 052316 000406          BR       $TYPEX      ;;EXIT
9477 052320 122766 000012 000002 1$:  (MPB  #LF,2(SP)    ;;BRANCH IF
9478 052326 001402          BEQ      $TYPEX      ;;<LF>
9479 052330 105227          INCB    (PC)+       ;;INC SPACE
9480 052332 000000          $CHARCNT: .WORD 0    ;;COUNT
9481 052334 000207          $TYPEX: RTS      PC
9482
9483
9484
9485
9486
9487
9488
9489
9490
9491
9492
9493
9494
9495
9496
9497
9498
9499
9500
9501
9502
9503
9504
9505
9506
9507
9508
9509
9510 052336 017646 000000          $TYPOS: MOV      @ (SP),-(SP)    ;;PICKUP THE MODE
9511 052342 116637 000001 052561  MOV      1(SP), $OFILL    ;;LOAD ZERO FILL SWITCH
9512 052350 112637 052563          MOV      (SP)+, $OMODE+1  ;;NUMBER OF DIGITS TO TYPE
9513 052354 062716 000002          ADD      #2, (SP)        ;;ADJUST RETURN ADDRESS
9514 052360 000406          BR       $TYPON
9515 052362 112737 000001 052561  $TYPOC: MOV      #1, $OFILL    ;;SET THE ZERO FILL SWITCH
9516 052370 112737 000006 052563  MOV      #6, $OMODE+1    ;;SET FOR SIX(6) DIGITS
9517 052376 112737 000005 052560  $TYPON: MOV      #5, $OCNT    ;;SET THE ITERATION COUNT
9518 052404 010346          MOV      R3, -(SP)      ;;SAVE R3
9519 052406 010446          MOV      R4, -(SP)      ;;SAVE R4
9520 052410 010546          MOV      R5, -(SP)      ;;SAVE R5
9521 052412 113704 052563          MOV      $OMODE+1, R4   ;;GET THE NUMBER OF DIGITS TO TYPE
9522 052416 005404          NEG      R4
9523 052420 062704 000006          ADD      #6, R4         ;;SUBTRACT IT FOR MAX. ALLOWED
9524 052424 110437 052562          MOV      R4, $OMODE     ;;SAVE IT FOR USE
9525 052430 113704 052561          MOV      $OFILL, R4     ;;GET THE ZERO FILL SWITCH
9526 052434 016605 000012          MOV      12(SP), R5     ;;PICKUP THE INPUT NUMBER
9527 052440 005003          CLR      R3            ;;CLEAR THE OUTPUT WORD
9528 052442 006105          1$:  ROL      R5        ;;ROTATE MSB INTO 'C'
9529 052444 000404          BR       3$            ;;GO DO MSB
    
```

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE
 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 ;*\$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 ;*CALL:
 ;* MOV NUM, -(SP) ;;NUMBER TO BE TYPED
 ;* TYPOS ;;CALL FOR TYPEOUT
 ;* .BYTE N ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 ;* .BYTE M ;;M=1 OR 0
 ;* ;;1=TYPE LEADING ZEROS
 ;* ;;0-SUPPRESS LEADING ZEROS

;*\$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 ;*\$TYPOS OR \$TYPOC
 ;*CALL:
 ;* MOV NUM, -(SP) ;;NUMBER TO BE TYPED
 ;* TYPON ;;CALL FOR TYPEOUT

;*\$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 ;*CALL:
 ;* MOV NUM, -(SP) ;;NUMBER TO BE TYPED
 ;* TYPOC ;;CALL FOR TYPEOUT

```

9530 052446 006105      2$:  ROL    R5          ;;FORM THIS DIGIT
9531 052450 006105      ROL    R5
9532 052452 006105      ROL    R5
9533 052454 010503      MOV    R5,R3
9534 052456 006103      3$:  ROL    R3          ;;GET LSB OF THIS DIGIT
9535 052460 105337 052562  DECB   $OMODE        ;;TYPE THIS DIGIT?
9536 052464 100016      BPL    7$            ;;BR IF NO
9537 052466 042703 177770  BIC    #177770,R3    ;;GET RID OF JUNK
9538 052472 001002      BNE    4$            ;;TEST FOR 0
9539 052474 005704      TST    R4            ;;SUPPRESS THIS 0?
9540 052476 001403      BEQ    5$            ;;BR IF YES
9541 052500 005204      4$:  INC    R4          ;;DON'T SUPPRESS ANYMORE 0'S
9542 052502 052703 000060  BIS    #'0,R3        ;;MAKE THIS DIGIT ASCII
9543 052506 052703 000040  5$:  BIS    #' ,R3      ;;MAKE ASCII IF NOT ALREADY
9544 052512 110337 052556  MOVB   R3,8$         ;;SAVE FOR TYPING
9545 052516 104400 052556  TYPE   ,8$           ;;GO TYPE THIS DIGIT
9546 052522 105337 052560  7$:  DECB   $OCNT       ;;COUNT BY 1
9547 052526 003347      BGT    2$            ;;BR IF MORE TO DO
9548 052530 002402      BLT    6$            ;;BR IF DONE
9549 052532 005204      INC    R4            ;;INSURE LAST DIGIT ISN'T A BLANK
9550 052534 000744      BR     2$            ;;GO DO THE LAST DIGIT
9551 052536 012605      6$:  MOV    (SP)+,R5    ;;RESTORE R5
9552 052540 012604      MOV    (SP)+,R4    ;;RESTORE R4
9553 052542 012603      MOV    (SP)+,R3    ;;RESTORE R3
9554 052544 016666 000002 000004  MOV    2(SP),4(SP)  ;;SET THE STACK FOR RETURNING
9555 052552 012616      MOV    (SP)+,(SP)
9556 052554 000002      RTI                    ;;RETURN
9557 052556      000      8$:  .BYTE  0            ;;STORAGE FOR ASCII DIGIT
9558 052557      000      .BYTE  0            ;;TERMINATOR FOR TYPE ROUTINE
9559 052560      000      $OCNT: .BYTE  0        ;;OCTAL DIGIT COUNTER
9560 052561      000      $OFILL: .BYTE  0      ;;ZERO FILL SWITCH
9561 052562 000000      $OMODE: .WORD  0      ;;NUMBER OF DIGITS TO TYPE
9562
9563      ;;*****
9564
9565      .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
9566
9567      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
9568      ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
9569      ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
9570      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
9571      ;*REPLACED WITH SPACES.
9572      ;*CALL:
9573      ;*      MOV    NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
9574      ;*      TYPDS                    ;;GO TO THE ROUTINE
9575
9576      $TYPDS:
9577      MOV    R0,-(SP)      ;;PUSH R0 ON STACK
9578      MOV    R1,-(SP)      ;;PUSH R1 ON STACK
9579      MOV    R2,-(SP)      ;;PUSH R2 ON STACK
9580      MOV    R3,-(SP)      ;;PUSH R3 ON STACK
9581      MOV    R5,-(SP)      ;;PUSH R5 ON STACK
9582      MOV    #2020,-(SP)    ;;SET BLANK SWITCH AND SIGN
9583      MOV    20(SP),R5     ;;GET THE INPUT NUMBER
9584      BPL    1$            ;;BR IF INPUT IS POS.
9585      NEG    R5            ;;MAKE THE BINARY NUMBER POS.

```

```

9586 052612 112766 000055 000001      MOVB  #'-.1(SP)      ;;MAKE THE ASCII NUMBER NEG.
9587 052620 005000                    1$: CLR  R0            ;;ZERO THE CONSTANTS INDEX
9588 052622 012703 053000              MOV  #$DBLK,R3      ;;SETUP THE OUTPUT POINTER
9589 052626 112723 000040              MOVB #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
9590 052632 005002                    2$: CLR  R2            ;;CLEAR THE BCD NUMBER
9591 052634 016001 052770              MOV  $DTBL(R0),R1  ;;GET THE CONSTANT
9592 052640 160105                    3$: SUB  R1,R5        ;;FORM THIS BCD DIGIT
9593 052642 002402                    BLT  4$            ;;BR IF DONE
9594 052644 005202                    INC  R2            ;;INCREASE THE BCD DIGIT BY 1
9595 052646 000774                    BR   3$
9596 052650 060105                    4$: ADD  R1,R5        ;;ADD BACK THE CONSTANT
9597 052652 005702                    TST  R2            ;;CHECK IF BCD DIGIT=0
9598 052654 001002                    BNE  5$            ;;FALL THROUGH IF 0
9599 052656 105716                    TSTB (SP)          ;;STILL DOING LEADING 0'S?
9600 052660 100407                    BMI  7$            ;;BR IF YES
9601 052662 106316                    5$: ASLB (SP)        ;;MSD?
9602 052664 103003                    BCC  6$            ;;BR IF NO
9603 052666 116663 000001 177777      MOVB 1(SP),-1(R3)  ;;YES--SET THE SIGN
9604 052674 052702 000060              BIS  #'0,R2        ;;MAKE THE BCD DIGIT ASCII
9605 052700 052702 000040              6$: BIS  #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
9606 052704 110223                    7$: MOVB R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
9607 052706 005720                    TST  (R0)+         ;;JUST INCREMENTING
9608 052710 020027 000010              CMP  R0,#10        ;;CHECK THE TABLE INDFX
9609 052714 002746                    BLT  2$            ;;GO DO THE NEXT DIGIT
9610 052716 003002                    BGT  8$            ;;GO TO EXIT
9611 052720 010502                    MOV  R5,R2         ;;GET THE LSD
9612 052722 000764                    BR   6$            ;;GO CHANGE TO ASCII
9613 052724 105726                    8$: TSTB (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
9614 052726 100003                    BPL  9$            ;;BR IF NO
9615 052730 116663 177777 177776      9$: MOVB -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
9616 052736 105013                    CLRB (R3)          ;;SET THE TERMINATOR
9617 052740 012605                    MOV  (SP)+,R5      ;;POP STACK INTO R5
9618 052742 012603                    MOV  (SP)+,R3      ;;POP STACK INTO R3
9619 052744 012602                    MOV  (SP)+,R2      ;;POP STACK INTO R2
9620 052746 012601                    MOV  (SP)+,R1      ;;POP STACK INTO R1
9621 052750 012600                    MOV  (SP)+,R0      ;;POP STACK INTO R0
9622 052752 104400 053000              TYPE , $DBLK      ;;NOW TYPE THE NUMBER.
9623 052756 016666 000002 000004      MOV  2(SP),4(SP)  ;;ADJUST THE STACK
9624 052764 012616                    MOV  (SP)+,(SP)
9625 052766 000002                    RTI                ;;RETURN TO USER
9626 052770 023420                    $DTBL: 10000.
9627 052772 001750                    1000.
9628 052774 000144                    100.
9629 052776 000012                    10.
9630 053000 000004                    $DBLK: .BLKW 4
9631
9632
9633
9634
9635
9636
9637
9638
9639
9640
9641
    ;:*****
    .SBTTL RANDOM NUMBER GENERATOR ROUTINE
    ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
    ;*WITH A RANGE OF 0 TO 2(+33)-1.
    ;*CALL:
    ;*   JSR PC,$RAND    ;;CALL THE ROUTINE
    ;*   RETURN          ;;RETURN HERE THE RANDOM
    ;*                   ;;NUMBER WILL BE IN
    
```

```

9642          ;*                               :: $HINUM,$LONUM
9643
9644 053010    $RAND:                          MOV     R0,-(SP)           :: PUSH R0 ON STACK
9645 053010    010046                          MOV     R1,-(SP)           :: PUSH R1 ON STACK
9646 053012    010146                          MOV     R2,-(SP)           :: PUSH R2 ON STACK
9647 053014    010246                          MOV     $LONUM,R0          :: SET R0 WITH LOW
9648 053016    01370C    053110              MOV     $HINUM,R1         :: SET R1 WITH HIGH
9649 053022    013701    053106              MOV     #-7,R2           :: SET SHIFT COUNT
9650 053026    012702    177771          1$:    ASL     R0               :: SHIFT R0 LEFT AND
9651 053032    006300                          ROL     R1                 :: ROTATE CARRY INTO R1 AND
9652 053034    006101                          INC     R2                 :: CHECK FOR DONE
9653 053036    005202                          BNE    1$                 :: CONTINUE SHIFT LOOP
9654 053040    001374                          ADD     $LONUM,R0         :: ADD NUMBER TO MAKE X 129
9655 053042    063700    053110              ADC     R1                 :: PROPOGATE CARRY
9656 053046    005501                          ADD     $HINUM,R1        :: ADD NUMBER TO MAKE X 129
9657 053050    063701    053106              ADD     #1057,R0         :: ADD LOW CONSTANT
9658 053054    062700    001057              ADC     R1                 :: PROPOGATE CARRY
9659 053060    005501                          ADD     #47401,R1        :: ADD HIGH CONSTANT
9660 053062    062701    047401              MOV     R0,$LONUM        :: SAVE R0
9661 053066    010037    053110              MOV     R1,$HINUM        :: SAVE R1
9662 053072    010137    053106              MOV     (SP)+,R2         :: POP STACK INTO R2
9663 053076    012602                          MOV     (SP)+,R1         :: POP STACK INTO R1
9664 053100    012601                          MOV     (SP)+,R0         :: POP STACK INTO R0
9665 053102    012600                          RTS     PC                 :: RETURN
9666 053104    000207
9667 053106    176543    $HINUM: .WORD 176543
9668 053110    123456    $LONUM: .WORD 123456
9669
9670          ;:*****
9671
9672          .SBTTL TRAP DECODER
9673
9674          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
9675          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
9676          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
9677          ;*GO TO THAT ROUTINE.
9678
9679 053112    010046    $TRAP: MOV     R0,-(SP)           :: SAVE R0
9680 053114    016600    000002              MOV     2(SP),R0         :: GET TRAP ADDRESS
9681 053120    005740                          TST     -(R0)            :: BACKUP BY 2
9682 053122    111000                          MOVB   (R0),R0           :: GET RIGHT BYTE OF TRAP
9683 053124    016000    053132              MOV     $TRPAD(R0),R0    :: INDEX TO TABLE
9684 053130    000200                          RTS     R0                :: GO TO ROUTINE
9685
9686          .SBTTL TRAP TABLE
9687
9688          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
9689          ;*BY THE 'TRAP' INSTRUCTION.
9690
9691          :
9692          : ROUTINE
9693          : -----
9694 053132    $TRPAD:
9695 053132    052122    $TYPE  ::CALL=TYPE      TRAP+0(104400) TTY TYPEOUT ROUTINE
9696 053134    052362    $TYPOC ::CALL=TYPOC     TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
9697 053136    052336    $TYPOS ::CALL=TYPOS     TRAP+4(104404) TYPE OCTAL NUMBER (NO LEADING ZEROS)
    
```


9698	053140	052376	\$TYPON	::CALL=TYPON	TRAP+6(104406)	TYPE OCTAL NUMBER (AS PER LAST CALL)
9699	053142	052564	\$TYPDS	::CALL=TYPDS	TRAP+10(104410)	TYPE DECIMAL NUMBER (WITH SIGN)
9700	053144	052026	\$SAVREG	::CALL=SAVREG	TRAP+12(104412)	SAVE R0-R5 ROUTINE
9701	053146	052064	\$RESREG	::CALL=RESREG	TRAP+14(104414)	RESTORE R0-R5 ROUTINE
9702						
9703	053150	054232	CLEAN	::CALL=RSET	TRAP+16(104416)	GO RESET ALL REGISTERS.
9704	053152	054202	ABORTT	::CALL=SKIPT	TRAP+20(104420)	THIS WILL SKIP TO THE NEXT TEST
9705	053154	054646	MMDDES	::CALL=MMSKIP	TRAP+22(104422)	IF SWITCH # IS ON SKIP TO THE NEXT TEST
9706	053156	054670	MSIZER	::CALL=SIZE	TRAP+24(104424)	DETERMINE THE HIGHEST ADDRESS IN MEMORY
9707	053160	054322	SKBADR	::CALL=SKPBAD	TRAP+26(104426)	SKIP TEST IF ERROR ADDRESS REGISTER IS I
9708	053162	054346	SKBERR	::CALL=SKPBER	TRAP+30(104430)	SKIP TEST IF ERROR REGISTER IS INOPERA
9709	053164	054364	SKBCNR	::CALL=SKPBCN	TRAP+32(104432)	SKIP TEST IF CONTROL REGISTER IS INOPERA
9710	053166	054402	SKBMNR	::CALL=SKPBMN	TRAP+34(104434)	SKIP TEST IF MAINTENANCE REGISTER IS INO
9711	053170	054420	SKBHMR	::CALL=SKPBHM	TRAP+36(104436)	SKIP TEST IF HIT/MISS REGISTER IS IN OPE
9712	053172	041606	RANDWR	::CALL=WRRAND	TRAP+40(104440)	FILL BUFFER WITH RANDOM SEQUENCE
9713						
9714	053174	057436	RS4HAN	::CALL=CALRS4	TRAP+42(104442)	DO RS04 FUNCTION
9715	053176	056466	RP4HAN	::CALL=CALRP4	TRAP+44(104444)	DO RP04 FUNCTION
9716	053200	062140	RH4HAN	::CALL=CALRH4	TRAP+46(104446)	DO MBT FUNCTION
9717	053202	060372	RK5HAN	::CALL=CALRK5	TRAP+50(104450)	DO RK05 FUNCTION
9718	053204	061406	UBEHAN	::CALL=CALUBE	TRAP+52(104452)	DO UBE FUNCTION

::*****

.SBTTL POWER DOWN AND UP ROUTINES

:POWER DOWN ROUTINE

9724	053206	012737	053334	000024	\$PWRDN: MOV	#\$ILLUP,@#PWRVEC	::SET FOR FAST UP
9725	053214	012737	000340	000026	MOV	#340,@#PWRVEC+2	::PRIO:7
9726	053222	010046			MOV	R0,-(SP)	::PUSH R0 ON STACK
9727	053224	010146			MOV	R1,-(SP)	::PUSH R1 ON STACK
9728	053226	010246			MOV	R2,-(SP)	::PUSH R2 ON STACK
9729	053230	010346			MOV	R3,-(SP)	::PUSH R3 ON STACK
9730	053232	010446			MOV	R4,-(SP)	::PUSH R4 ON STACK
9731	053234	010546			MOV	R5,-(SP)	::PUSH R5 ON STACK
9732	053236	010637	053340		MOV	SP,\$SAVR6	::SAVE SP
9733	053242	012737	053254	000024	MOV	#\$PWRUP,@#PWRVEC	::SET UP VECTOR
9734	053250	000000			HALT		
9735	053252	000776			BR	.-2	::HANG UP

:POWER UP ROUTINE

9738	053254	013706	053340		\$PWRUP: MOV	\$SAVR6,SP	::GET SP
9739	053260	005037	053340		CLR	\$SAVR6	::WAIT LOOP FOR THE TTY
9740	053264	005237	053340		1\$: INC	\$SAVR6	::WAIT FOR THE INC
9741	053270	001375			BNE	1\$::OF WORD
9742	053272	012605			MOV	(SP)+,R5	::POP STACK INTO R5
9743	053274	012604			MOV	(SP)+,R4	::POP STACK INTO R4
9744	053276	012603			MOV	(SP)+,R3	::POP STACK INTO R3
9745	053300	012602			MOV	(SP)+,R2	::POP STACK INTO R2
9746	053302	012601			MOV	(SP)+,R1	::POP STACK INTO R1
9747	053304	012600			MOV	(SP)+,R0	::POP STACK INTO R0
9748	053306	012737	053206	000024	MOV	#\$PWRDN,@#PWRVEC	::SET UP THE POWER DOWN VECTOR
9749	053314	012737	000340	000026	MOV	#340,@#PWRVEC+2	::PRIO:7
9750	053322	104400			TYPE		::REPORT THE POWER FAILURE
9751	053324	065031			\$PWRMG: .WORD	POWERM	::POWER FAIL MESSAGE POINTER
9752	053326	012716			MOV	(PC)+,(SP)	::RESTART AT START
9753	053330	004112			\$PWRAD: .WORD	START	::RESTART ADDRESS

```

9754 053332 000002
9755 053334 000000
9756 053336 000776
9757 053340 000000
9758
9759
9760
9761
9762
9763
9764
9765
9766
9767
9768
9769
9770
9771
9772
9773
9774
9775
9776
9777
9778
9779
9780
9781 053342 010046
9782 053344 010146
9783 053346 010246
9784 053350 010346
9785 053352 013746 000004
9786 053356 013746 000006
9787 053362 013746 000114
9788 053366 013746 000116
9789 053372 010600
9790 053374 013737 177776 000006
9791 053402 012701 003776
9792 053406 105727
9793 053410 000200
9794 053412 100065
9795 053414 012737 053560 000004
9796 053422 005737 177572
9797 053426 052737 100000 053410
9798 053434 005046
9799 053436 012702 172340
9800 053442 012703 000010
9801 053446 012762 077406 177740 1$:
9802 053454 011622
9803 053456 062716 000200
9804 053462 077307
9805 053464 012742 177600
9806 053470 005042
9807 053472 012737 053510 000004
9808 053500 012737 000020 172516
9809 053506 000401

```

```

RTI
$ILLUP: HALT          ;; THE POWER UP SEQUENCE WAS STARTED
BR      .-2          ;; BEFORE THE POWER DOWN WAS COMPLETE
$SAVR6: 0           ;; PUT THE SP HERE
;:*****
.SBTTL ROUTINE TO SIZE MEMORY
; *CALL:
; *   JSR    PC,$SIZE
; *   RETURN
; *$LSTAD WILL CONTAIN:
; *   WITH KT11 OPTION      -- LAST VIRTUAL ADDRESS OF THE LAST BANK
; *   WITHOUT KT11 OPTION  -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
; *$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
; *$KT11 IS THE MEMORY MANAGEMENT KEY
; *$BIT07 = 0 DON'T USE MEMORY MANAGEMENT
; *       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.
$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    @#ERRVEC,-(SP) ;;SAVE PRESENT ERROR VECTOR PS & PC
        MOV    @#ERRVEC+2,-(SP)
        MOV    @#114,-(SP)   ;;SAVE PRESENT PARITY VECOT PS & PC
        MOV    @#116,-(SP)
        MOV    SP,R0         ;;SAVE THE STACK POINTER
        MOV    @#PS,@#ERRVEC+2 ;;SET ERRVEC PS TO PRESENT PS
        MOV    #3776,R1      ;;SETUP ADDRESS
        TSTB   (PC)+         ;;USE MEMORY MANAGEMENT?
$KT11:  .WORD  200          ;;SET TO USE MEMORY MANAGEMENT
        BPL    $CORE        ;;BR IF NO
        MOV    # $KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
        TST    @#SR0        ;;KT11 ARE YOU THERE?
        BIS    #100000,$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$,@#ERRVEC ;;CATCH TIMEOUT IF NO SR3
        MOV    #20,@#SR3    ;;ENABLE 22-BIT ADDRESSING
        BR     3$          ;;THIS PDP-11 HAS A SR3 REG.

```

```

9810 053510 022626          2$:  CMP      (SP)+,(SP)+      ;;CLEAN OFF THE STACK--NO SR3.
9811 053512 005237 177572  3$:  INC      @#SR0              ;;TURN ON MEMORY MANAGEMENT
9812 053516 012737 053550 000004  MOV     #SKTOUT,@#ERRVEC    ;;SET FOR TIME OUT
9813 053524 012737 053672 000114  MOV     #SMTMOUT,@#114     ;;SET FOR MEM REF TIMEOUT
9814 053532 005737 143776  4$:  TST     @#143776          ;;TRAP ON NON-EX-MEM
9815 053536 062712 000040  ADD     #40,(R2)           ;;MAKE A 1K STEP
9816 053542 023712 172356  CMP     @#KIPAR7,(R2)      ;;LAST ONE?
9817 053546 101371  4$:  BHI     4$                 ;;NO--TRY IT
9818 053550 011202  SKTOUT: MOV    (R2),R2          ;;GET LAST BANK+1
9819 053552 005037 177572  CLR     @#SR0              ;;TURN OFF MEMORY MANAGEMENT
9820 053556 000421  BR      $SIZEX
9821 053560 042737 100000 053410 SKTNEX: BIC    #100000,$KT11   ;;KT11 NON-EXISTENT
9822 053566 012737 053616 000004 SCORE: MOV    #SCROUT,@#ERRVEC  ;;SET FOR TIMEOUT
9823 053574 005002  CLR     R2                 ;;SET UP BANK
9824 053576 062701 004000  1$:  ADD     #4000,R1          ;;INCREMENT BY 1K
9825 053602 062702 000040  ADD     #40,R2             ;;1K STEP
9826 053606 005711  TST     (R1)               ;;TRAP ON TIME OUT
9827 053610 022701 177776  CMP     #177776,R1         ;;LAST ONE
9828 053614 001370  BNF     1$                 ;;NO--TRY AGAIN
9829 053616 162701 004000  SCROUT: SUB   #4000,R1
9830 053622 162702 000040  $SIZEX: SUB   #40,R2        ;;DROP BACK
9831 053626 010006  MOV     R0,SP              ;;RESTORE THE STACK
9832 053630 012637 000116  MOV     (SP)+,@#116        ;;RESTOR PARITY VECTOR
9833 053634 012637 000114  MOV     (SP)+,@#114
9834 053640 012637 000006  MOV     (SP)+,@#ERRVEC+2  ;;RESTORE ERROR VECTOR
9835 053644 012637 000004  MOV     (SP)+,@#ERRVEC
9836 053650 010137 053724  MOV     R1,$LSTAD          ;;LAST ADDRESS
9837 053654 010237 053726  MOV     R2,$LSTBK          ;;LAST BANK
9838 053660 012603  MOV     (SP)+,R3           ;;RESTORE R3
9839 053662 012602  MOV     (SP)+,R2           ;;RESTORE R2
9840 053664 012601  MOV     (SP)+,R1           ;;RESTORE R1
9841 053666 012600  MOV     (SP)+,R0           ;;RESTORE R0
9842 053670 000207  RTS     FR
9843 053672 032737 000001 177744 SMTMOUT: BIT   #BIT0,@#MEMERR  ;;MAKE SURE TRAP TO 114 IS DUE
9844 053700 001005  BNE     1$                 ;;TO MEMORY REFERENCE TIMEOUT
9845 053702 032737 100000 177744  BIT     #BIT15,@#MEMERR    ;;IF NOT, IS IT AN ABORT?.
9846 053710 001001  BNE     1$                 ;;CPU ABORT?
9847 053712 000002  RTI     1$                 ;;IF YES, EXIT OUT
9848 053714 012737 177777 177744  1$:  MOV     #-1,@#MEMERR      ;;IF NOT, CONTINUE
9849 053722 000712  BR      SKTOUT             ;;CLEAR THE MEM ERROR REG
9850 053724 000000  $LSTAD: .WORD 0              ;;CONTAINS THE LAST ADDRESS
9851 053726 000000  $LSTBK: .WORD 0              ;;CONTAINS THE LAST BANK

```

9853
9854
9855
9856
9857
9858
9859
9860
9861
9862
9863
9864
9865

;;*****
;SBTTL DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

;;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
;;*UNSIGNED OCTAL ASCII NUMBER.

;;*CALL

```

;;*  MOV     #PNTR,-(SP)      ;;POINTER TO LOW WORD OF BINARY NUMBER
;;*  JSR     PC,@#SDB20      ;;CALL THE ROUTINE
;;*  RETURN                  ;;THE ADDRESS OF THE FIRST ASCII CHAR. IS ON THE STACK

```

```

9866 053730 104412          $DB20: SAVREG          ;;SAVE ALL REGISTERS
9867 053732 016601 000002  MOV      2(SP),R1      ;;PICKUP THE POINTER TO LOW WORD
9868 053736 012705 054047  MOV      #SOCTLV+13.,R5 ;;POINTER TO DATA TABLE
9869 053742 012704 000014  MOV      #12.,R4       ;;DO ELEVEN CHARACTERS
9870 053746 012703 177770  MOV      #^C7,R3       ;;MASK
9871 053752 012100  MOV      (R1)+,R0      ;;LOWER WORD
9872 053754 012101  MOV      (R1)+,R1      ;;HIGH WORD
9873 053756 005002  CLR      R2           ;;TERMINATOR
9874 053760 110245 1$: MOVB   R2,-(R5)     ;;PUT CHARACTER IN DATA TABLE
9875 053762 010002  MOV      R0,R2        ;;GET THIS DIGIT
9876 053764 005304  DEC      R4           ;;COUNT THIS CHARACTER
9877 053766 003007  BGT     3$           ;;BR IF NOT THE LAST DIGIT
9878 053770 001405  BEQ     2$           ;;BR IF IT IS THE LAST DIGIT
9879 053772 005205  INC     R5           ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
9880 053774 010566 000002  MOV     R5,2(SP)     ;;ASCIZ CHAR. & PUT IT ON THE STACK
9881 054000 104414  RESREG          ;;RESTORE ALL REGISTERS
9882 054002 000207  RTS     PC           ;;RETURN TO USER
9883 054004 006203 2$: ASR   R3           ;;POSITION THE MASK FOR THE LAST DIGIT
9884 054006 006001 3$: ROR   R1           ;;POSITION THE BINARY NUMBER FOR
9885 054010 006000  ROR   R0           ;;
9886 054012 006001  ROR   R1           ;;
9887 054014 006000  ROR   R0           ;;
9888 054016 006001  ROR   R1           ;;
9889 054020 006000  ROR   R0           ;;
9890 054022 040302  BIC   R3,R2        ;;MASK OUT ALL JUNK
9891 054024 062702 000060  ADD   #'0,R2       ;;MAKE THIS CHAR. ASCII
9892 054030 000753  BR    1$           ;;GO PUT IT IN THE DATA TABLE
9893 054032 000016  $OCTLV: .BLKB 14.  ;;RESERVE DATA TABLE
9894
9895
9896
9897
9898 054050 011637 001630  ;THIS ROUTINE IS CALLED BY UNEXPECTED TRAPS TO VECTOR ERRVEC.
9899 054054 012737 054072 001632  ;THE ERROR IS REPORTED AND CONTROL IS TRANSFERRED BACK TO THE TEST
9900 054062 013737 177766 001634  ;FOLLOWING THE ONE THAT WAS INTERRUPTED WHEN THE ERROR OCCURRED.
9901 054070 022626  CPSPUR: MOV    (SP),$TMP1
9902 054072 104150  MOV    #1$, $TMP2
9903 054074 104420  MOV    @#CPUERR,$TMP3
9904
9905 1$: CMP    (SP)+,(SP)+  ;;RESET THE STACK
9906 054076 012737 054174 000114  ERROR 150
9907 054104 013700 177744  SKIPT
9908 054110 032700 000014  ;THIS ROUTINE HANDLE UNEXPECTED RAPS TO #CACHVEC.
9909 054114 001405  SPUR: MOV    #10$,@#CACHVEC
9910 054116 013701 177740  MOV    @#MEMERR,R0
9911 054122 042701 176000  BIT    #14,R0        ;;SEE IF IT WAS A MAIN MEMORY ERROR.
9912 054126 005711  BEQ    9$
9913 054130 012737 054076 000114  MOV    @#LOADRS,R1   ;;IF SO THERE IS BAD PARITY IN THE
9914 054136 013737 177744 001636  BIC    #176000,R1   ;;CACHE AND IT MUST BE PURGED...
9915 054144 013737 177740 001630  TST   (R1)
9916 054152 013737 177742 001632 9$: MOV    #SPUR,@#CACHVEC
9917 054160 011637 001634  MOV    @#MEMERR,$TMP4 ;;TRAP HERE IF AN UNEXPECTED
9918 054164 022626  MOV    @#LOADRS,$TMP1 ;;ERROR, PARITY, OCCURS.
9919 054166 104014  MOV    @#HIADRS,$TMP2
9920 054170 000005  MOV    (SP),$TMP3
9921 054172 104420  CMP    (SP)+,(SP)+
          ERROR 14
          RESET
          SKIPT
          ;;TO STOP THE ACTION OF ANY I/O DEVICE....
          ;?????
    
```

```

9922 054174 022626      10$:  CMP      (SP)+,(SP)+
9923 054176 000137 054130      JMP      9$
9924
9925      ;THIS ROUTINE IS CALLED BY THE TRAP CATCHER CALL SKIPT.
9926      ;IT TELLS THE USER THAT THE CURFENT TEST HAS BEEN
9927      ;ABORTED AND THAT CONTROL IS BEING PASSED TO THE NEXT TEST.
9928 054202 011637 001630      ABORTT: MOV      (SP), $TMP1
9929 054206 112737 000015 001516      MOVB     #15, $ITEMB
9930 054214 022626      CMP      (SP)+,(SP)+
9931 054216 004737 054744      JSR     PC, ERTYPE
9932 054222 104416      RSET
9933 054224 000177 000000      JMP     @SKAD      ;GO TO @SKAD, WHICH SHOULD
9934      ;BE SET TO THE
9935 054230 000000      SKAD:  .WORD    0      ;ADDRESS OF THE NEXT TEST.
9936
9937
9938      ;THIS ROUTINE IS CALLED BY THE TRAP CATCHER CALL RSET. IT CLEARS ALL
9939      ;THE IMPORTANE REGISTERS AND RESETS THE STACK.
9940 054232      CLEAN:
9941
9942 054232 012737 054076 000114      MOV     #SPUR, @CACHVEC
9943 054240 012737 054050 000004      MOV     #CPSPUR, @ERRVEC
9944 054246 011637 054320      MOV     (SP), BACKAD
9945 054252 012706 001500      MOV     #STACK, SP
9946 054256 005037 177750      CLR     @MAINT      ;CLEAR ALL CONTROL AND ERROR
9947 054262 005037 177572      CLR     @MMR0      ;REGISTERS.
9948 054266 005037 172516      CLR     @MMR3
9949 054272 005037 177746      CLR     @CONTRL
9950 054276 012737 177777 177744      MOV     #-1, @MEMERR
9951 054304 005037 177766      CLR     @CPUERR
9952 054310 005037 177776      CLR     @PSW
9953 054314 000177 000000      JMP     @BACKAD
9954 054320 000000      BACKAD: .WORD    0
9955
9956
9957      ;COME HERE TO TEST THE REGISTER FLAGS AND USE THEM TO DETERMINE WHETHER
9958      ;OR NOT TO SKIP A TEST WHICH RELIES ON THE FUNCTIONALLITY OF THAT REGISTER
9959      ;TO BE PROPERLY RUN.
9960      ;THESE ROUTINES ARE CALLED BY THE TRAP CATCHER CALLS:
9961      :      SKPBAD      SKIPT IF BAD ERROR ADDRESS REGISTER
9962      :      SKPBER      SKIPT IF BAD ERROR REGISTER
9963      :      SKPBCN      SKIPT IF BAD CONTROL REGISTER
9964      :      SKPBMN      SKIPT IF BAD MAINTENANCE REGISTER
9965      :      SKPBHM      SKIPT IF BAD HIT/MISS REGISTER
9966      :
9967
9968 054322 005737 054440      SKBADR: TST     _OAFLG
9969 054326 001004      BNE     1$
9970 054330 005737 054442      TST     HIAFLG
9971 054334 001001      BNE     1$
9972 054336 000002      RTI
9973 054340 104400      1$:  TYPE
9974 054342 066013      .WORD  ADRNG
9975 054344 000433      BR     SKRNG
9976
9977 054346 005737 054444      SKBERR: TST     MMRFLG
    
```

```

9978 054352 001001          BNE      1$
9979 054354 000002          RTI
9980 054356 104400      1$:      TYPE
9981 054360 066123          .WORD   ERRNG
9982 054362 000424          BR       SKRNG
9983
9984 054364 005737 054446  SKBCNR: TST      CONFLG
9985 054370 001001          BNE      1$
9986 054372 000002          RTI
9987 054374 104400      1$:      TYPE
9988 054376 066223          .WORD   CNRNG
9989 054400 000415          BR       SKRNG
9990
9991 054402 005737 054450  SKBMNR: TST      MANFLG
9992 054406 001001          BNE      1$
9993 054410 000002          RTI
9994 054412 104400      1$:      TYPE
9995 054414 066325          .WORD   MNRNG
9996 054416 000406          BR       SKRNG
9997
9998 054420 005737 054452  SKBHMR: TST      HIMFLG
9999 054424 001001          BNE      1$
10000 054426 000002          RTI
10001 054430 104400      1$:      TYPE
10002 054432 066433          .WORD   HMRNG
10003
10004 054434 022626  SKRNG:  CMP      (SP)+,(SP)+      ;RESET THE STACK AND GO TO THE
10005 054436 104420          SKIPT      ;NEXT TEST.!!!
10006
10007 054440 000000          LOAFLG: .WORD   0      ;THESE ARE FLAGS USED TO DESIGNATE
10008 054442 000000          HIAFLG: .WORD   0      ;EITHER A GOOD OR A BAD REGISTER.
10009 054444 000000          MMRFLG: .WORD   0      ;GOOD WILL BE DESIGNATED BY A
10010 054446 000000          CONFLG: .WORD   0      ;0 BAD BY A NOT ZERO..
10011 054450 000000          MANFLG: .WORD   0
10012 054452 000000          HIMFLG: .WORD   0
10013 054454 000000          LOAFL2: .WORD   0
10014 054456 000000          HIAFL2: .WORD   0
10015 054460 000000          MMRFL2: .WORD   0
10016 054462 000000          CONFL2: .WORD   0
10017 054464 000000          MANFL2: .WORD   0
10018 054466 000000          HIMFL2: .WORD   0
10019
10020          ;THIS ROUTINE IS CALLED TO DETERMINE THE PARITY OF
10021          ;A DATA PATTERN. THE PATTERN WHICH IS TAKEN BY THIS
10022          ;ROUTINE AS ITS ARGUMENT SHOULD BE PUT IN R0. THEN
10023          ;TRANSFER CONTROL HERE BY EXECUTING:
10024          ;      JSR      PC,PARCNT
10025          ;WHEN THIS ROUTINE RETURNS THE NUMBER OF ON,(1), BITS
10026          ;IN R0 IS LEFT IN R2. THIS WOULD BE A NUMBER BETWEEN
10027          ;0 AND 16.
10028 054470 012701 000001  PARCNT: MOV      #1,R1
10029 054474 005002          CLR      R2
10030 054476 030100      1$:      BIT      R1,R0
10031 054500 001401          BEQ      2$
10032 054502 005202          INC      R2
10033 054504 006301      2$:      ASL      R1
    
```

```

10034 054506 103373          BCC 1$
10035 054510 000207          RTS  PC
10036
10037
10038
10039
10040
10041
10042
10043
10044
10045 054512 005037 177750
10046 054516 017700 125020
10047 054522 104416
10048 054524 005003
10049 054526 042700 000200
10050 054532 022700 000003
10051 054536 001032
10052 054540 104400
10053 054542 064766
10054 054544 012704 002734
10055 054550 012701 116710
10056 054554 012702 160000
10057 054560 012142
10058 054562 077402
10059 054564 012737 177777 054644
10060 054572 022703 125252
10061 054576 001001
10062 054600 000207
10063
10064
10065 054602 104400
10066 054604 064772
10067 054606 013737 054642 000060
10068
10069 054614 000000
10070 054616 012737 054512 000060
10071 054624 005077 124712
10072 054630 152777 000100 124702
10073 054636 000177 177366
10074 054642 000000
10075
10076 054644 177777
10077
10078
10079
10080
10081
10082
10083
10084
10085
10086
10087
10088 054646 032737 000200 177570
10089 054654 001001

          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,
;TURNS OUT TO BE A ^C (CONTROL-C) THEN MEMORY IS RESTORED. IF THE
;CHARACTER IS NOT ^C THEN A RETURN IS MADE TO THE TEST FOLLOWING
;THE ONE WHOSE EXECUTION WAS INTERRUPTED BY THE KEYBOARD INTERRUPT.
RESMON: CLR @MMANT
        MOV @STKB,R0
        RSET
        CLR R3
        BIC #BIT7,R0          ;GET THE CHARACTER, INITIALIZE THE REGISTERS
        CMP #3,R0           ;AND SEE IF THE CHARACTER WAS ^C.
        BNE NOCNC          ;BRANCH AND GO TO NEXT TEST IF NOT.
        TYPE NOCNC         ;ECHOE THE CONTROL-C AS '^C'
        .WORD CONCMS
CHAINQ: MOV #^D1500,R4          ;AND RESTORE THE MONITOR.
        MOV #BOTTOM+4,R1
        MOV #160000,R2
1$:     MOV (R1)+,-(R2)
        SOB R4,1$
        MOV #-1,MONF         ;RESET THE MONITOR RESTORED FLAG.
        CMP #125252,R3
        BNE STOP
        RTS  PC
        ;IF THE MONITOR WAS RESTORED BY THE
        ;.SEOP ROUTIN RETURN TO .SEOP.
        ;OTHERWISE HALT.
        ;TYPE THE MONITOR RESTORED MESSAGE.
STOP:   TYPE
        .WORD MMESRS
        MOV MONTTY,@TKVEC    ;SET THE TTY KEYBOARD INTERRUPT VECTOR
        ;TO ITS INITIAL STATE.
        ;AND HALT!!
        HALT
NOCNC:  CLR @STKB
        BISB #BIT6,@STKS    ;NOT CONTROL C SO RETURN TO NEXT TEST.
        JMP @SKAD          ;RETURN.
MONTTY: .WORD 0             ;STORAGE FOR THE TTY KEYBOARD VECTOR'S ORIGINAL
        ;CONTENTS.
MONF:   .WORD 177777       ;FLAG, IF NOT -1 THE MONITOR IS SAVED..

;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.
MMDES:  BIT #SW7,@SWR
        BNE 1$             ;IS THE SWITCH ON?
    
```

```

10090 054656 000002          RTI          ;NO, SO RETURN.
10091 054660 022626      1$:  CMP      (SP)+,(SP)+
10092 054662 104416          RSET
10093 054664 000177 177340  JMP      @SKAD      ;YES, GO TO THE NEXT TEST.
10094          ;THIS ROUTINE IS CALLED TO DETERMINE THE HIGHEST POSSIBLE
10095          ;ADDRESS IN MEMORY. IT IS CALLED THUS, BY TRAP CALL SIZE:
10096          ;
10097          ;
10098          ;
10099          ;
10100          ;
10101          ;
10102          ;
10103          ;
10104 054670 010046      MSIZER: MOV     R0,-(SP)      ;SAVE THE CONTENTS OF R0 AND R1
10105 054672 010146      MOV     R1,-(SP)      ;GET THE ADDRESS OF
10106 054674 016600 000004  MOV     4(SP),R0      ;THE CALL OF THE STACK.
10107 054700 013710 053726  MOV     $LSTBK,(R0)   ;GET THE ACCESSABLE BOUNDARY OF MEMORY
10108 054704 005060 000002  CLR     2(R0)
10109 054710 012701 000006  MOV     #6,R1        ;ROTATE THE 16-BIT 'BLOCK'
10110          ;NUMBER 6-BITS TO THE
10111 054714 006310      1$:  ASL     (R0)          ;LEFT AND TURN ON LOW ORDER
10112 054716 006160 000002  ROL     2(R0)        ;BITS 1-5 LEAVING BIT-0
10113 054722 077104      SOB     R1,1$        ;OFF SO AS TO CREATE
10114 054724 052710 000076  BIS     #76,(R0)     ;THE 22-BIT PHYSICAL ADDRESS OF
10115          ;THE HIGHEST WORD IN
10116          ;MEMORY.
10117 054730 022020      CMP     (R0)+,(R0)+  ;DETERMINE THE RETURN ADDRESS
10118 054732 010066 000004  MOV     R0,4(SP)     ;AND LEAVE ON THE STACK FOR
10119          ;AN RTI.
10120 054736 012601      MOV     (SP)+,R1     ;RESTORE R1 AND R0.
10121 054740 012600      MOV     (SP)+,R0
10122 054742 000002      RTI
10123          ;RETURN
10124          ;THIS ROUTINE IS USED TO TYPE AN ERROR MESSAGE
10125          ;WHICH IS IN THE DATA TABLE. IT IS CALLED BY
10126          ;THE $ERRRTE ROUTINE OR BY FIRST SETTING THE $ITEMB
10127          ;BYTE EQUAL TO THE ERROR TABLE ITEM NUMBER THAT IS
10128          ;TO BE PRINTED OUT AND THEN EXECUTING A JSR PC,ERTYPE
10128 054744 104400      ERTYPE: TYPE
10129 054746 001707      .WORD  $CRLF
10130 054750 010046      MOV     R0,-(SP)     ;SAVE R0
10131 054752 005000      CLR     R0
10132 054754 113700 001516  MOVB    $ITEMB,R0    ;GET THE ITEM NUMBER
10133 054760 001005      BNE     1$           ;ZERO?
10134 054762 013746 001520  MOV     $ERRPC,-(SP) ;YES, TYPE JUST THE PC
10135 054766 104402      TYPOC  ;OF THE ERROR CALL.
10136 054770 000137 055306  JMP     ERT5
10137          ;
10138 054774 005300      1$:  DEC     R0          ;MAKE R0 AN INDEX FOR THE
10139 054776 072027 000003  ASH     #3,R0        ;ERROR TABLE
10140 055002 062700 001716  ADD     # $ERRTB,R0
10141 055006 012037 055016  MOV     (R0)+,2$
10142 055012 001404      BEQ     3$           ;TYPE EM, ERROR MESSAGE.
10143 055014 104400      TYPE
10144 055016 000000      2$:  .WORD  0
10145 055020 104400      TYPE
    
```



```

10146 055022 001707          .WORD  $CRLF
10147 055024 012037 055034 3$:  MOV    (R0)+,4$      ;TYPE DH, DATA HEADER
10148 055030 001404          BEQ    5$
10149 055032 104400          TYPE
10150 055034 000000          4$:  .WORD  0
10151 055036 104400          TYPE
10152 055040 001707          .WORD  $CRLF
10153 055042 010146          5$:  MOV    R1,-(SP)      ;SAVE R1
10154 055044 012001          MOV    (R0)+,R1     ;GET DT, DATA TABLE ADDRESS
10155 055046 001002          BNE    6$
10156 055050 000137 055304    JMP    ERT4         ;JMP IF NO ERROR TABLE.
10157 055054 012000          6$:  MOV    (R0)+,R0     ;GET DF, DATA FORMAT ADDRESS
10158 055056 105710          ERT1: TSTB   (R0)       ;DATA FORMAT ENTRY EQUALS
10159 055060 001003          BNE    7$          ;ZERO?
10160 055062 013146          MOV    @ (R1)+,-(SP) ;YES, SO TYPE A 16-BIT
10161 055064 104402          TYPDC                ;OCTAL NUMBER
10162 055066 000500          BR     ERT2
10163 055070 122710 000001    7$:  CMPB   #1,(R0)     ;FORMAT EQUALS 1?
10164 055074 001003          BNE    8$
10165 055076 013146          MOV    @ (R1)+,-(SP) ;YES, TYPE A DECIMAL NUMBER
10166 055100 104410          TYPDS
10167 055102 000472          BR     ERT2
10168
10169 055104 122710 000002    8$:  CMPB   #2,(R0)     ;FORMAT 2?
10170 055110 001012          BNE    9$
10171 055112 012146          85$: MOV    (R1)+,-(SP) ;YES, TYPE A 22-BIT NUMBR
10172 055114 004737 053730    JSR    PC,$DB20    ;CALL $DB20 TO CONVERT THE
10173 055120 062716 000003    ADD    #3,(SP)     ;BINARY TO ASCII
10174 055124 012637 055132    MOV    (SP)+,29$   ;TYPE THE STRING
10175 055130 104400          TYPE
10176 055132 000000          29$: .WORD  0
10177 055134 000455          BR     ERT2
10178
10179 055136 122710 000004    9$:  CMPB   #4,(R0)     ;FORMAT 4?
10180 055142 001004          BNE    10$
10181 055144 013146          MOV    @ (R1)+,-(SP) ;YES, TYPE A 16-BIT
10182 055146 104404          TYPOS                ;OCTAL NUMBER SUPRESSING
10183 055150 016           .BYTE  16          ;LEADING ZEROES
10184 055151 000           .BYTE  0
10185 055152 000446          BR     ERT2
10186 055154 122710 000003    10$: CMPB   #3,(R0)     ;FORMAT 3?
10187 055160 001007          BNE    11$
10188 055162 013146          MOV    @ (R1)+,-(SP) ;YES CONVERT 16-BIT
10189 055164 012737 177777 055312 MOV    #-1,TVADFL  ;VIRTUAL ADDRESS TO 32-BIT
10190 055172 004737 055320    JSR    PC,TYPVAD   ;PHYSICAL ADDRESS AND TYPE
10191 055176 000434          BR     ERT2        ;RELOCATE ONLY IF SEG. IS ON.
10192 055200 122710 000005    11$: CMPB   #5,(R0)     ;FORMAT 5?
10193 055204 001005          BNE    12$
10194 055206 012137 055214    MOV    (R1)+,20$   ;PRINT ASCII STRING
10195 055212 104400          TYPE
10196 055214 000000          20$: .WORD  0
10197 055216 000426          BR     ERT3
10198
10199 055220 122710 000006    12$: CMPB   #6,(R0)     ;FORMAT 6
10200 055224 001005          BNE    13$
10201 055226 005037 055312    CLR    TVADFL
    
```

K 1

10202	055232	004737	055320		JSR	PC,TYPVAD		
10203	055236	000414			BR	ERT2		
10204								
10205	055240	122710	000007	13\$:	CMPB	#7,(R0)		;FORMAT 7?
10206	055244	001010			BNE	14\$		

```

10207 055246 012146          MOV      (R1)+,-(SP)
10208 055250 004737 053730   JSR      PC,$DB20
10209 055254 012637 055262   MOV      (SP)+,45$
10210 055260 104400          TYPE
10211 055262 000000          45$: .WORD 0
10212 055264 000401          BR      ERT2
10213
10214 055266 000000          14$: HALT          ;?????
10215
10216 055270 104400          ERT2: TYPE          ;PRINT A TAB AFTER TYPING AN
10217 055272 065076          .WORD $TAB          ;ERROR TABLE ENTRY OF ALL MODES
10218                          ;EXCEPT ASCII
10219 055274 005200          ERT3: INC      R0          ;POINT TO THE NEXT FORMAT BYTE
10220 055276 005711          TST      (R1)          ;IS THERE ANOTHER ENTRY?
10221 055300 001401          BEQ      ERT4
10222 055302 000665          BR      ERT1          ;YES, PROCESS IT
10223                          ;OTHERWISE:
10224 055304 012601          ERT4: MOV      (SP)+,R1          ;RESTORE R1
10225 055306 012600          ERT5: MOV      (SP)+,R0          ;RESTORE R0
10226 055310 000207          RTS      PC          ;AND RETURN
10227
10228 055312 000000          TVADFL: .WORD 0          ;FLAG USED TO TELL TYVAD
10229                          ;WHETHER TO CONDITIONALLY
10230                          ;OR UNCONDITIONALLY RELOCATE
10231                          ;WHEN TYPING AN ADDRESS,
10232                          ;-1 OR 0 RESPECTIVELY
10233
10234 055314 000000          TVADLO: .WORD 0          ;REGISTERS FOR THE 22-BIT
10235 055316 000000          TVADHI: .WORD 0          ;ADDRESS COMPUTED BY TYVAD.
10236
10237                          ;ROUTINE WHICH CONVERTS A 16-BIT ADDRESS TO A 22-BIT
10238                          ;ADDRESS. IF TVADFL IS -1, THEN CONVERT TO THE 22-BIT
10239                          ;REAL ADDRESS DEPENDENT ON SEG BEING ON OR OFF FOR RELOCATION.
10240                          ;IF TVADFL IS ZERO THEN UNCONDITIONAL USE THE KERNAL
10241                          ;PAR WHICH IS APPROPRIATE TO DO RELOCATION.
10242 055320 104412          TYPVAD: SAVREG
10243 055322 016601 000002          MOV      2(SP),R1          ;GET THE VIRTUAL
10244 055326 010137 055314          MOV      R1,TVADLO          ;ADDRESS
10245 055332 005037 055316          CLR      TVADHI
10246 055336 005737 055312          TST      TVADFL          ;CONDITIONALLY RELOCATE?
10247 055342 001404          BEQ      1$
10248 055344 032737 000001 177572          BIT      #1,@MMR0          ;YES, SEE IF MEMORY
10249 055352 001424          BEQ      2$          ;MANAGEMENT IS ON
10250 055354 005000          1$: CLR      R0          ;RELOCATE
10251 055356 073027 000003          ASHC    #3,R0          ;LEFT SHIFT R0 AND R1
10252 055362 006300          ASL      R0          ;THREE PLACES. R0 ONE
10253                          ;MORE SO THAT IT CONTAINS
10254                          ;2 X THE UPPER 3-BITS OF
10255 055364 000241          CLC
10256 055366 006001          ROR      R1          ;THE VIRTUAL ADDRESS
10257 055370 006001          ROR      R1          ;RESTORE R1 TO THE OFFSET
10258 055372 006001          ROR      R1          ;OF THE VIRTUAL ADDRESS
10259 055374 062700 172340          ADD      #KIPAR0,R0          ;TO THE PAR
10260                          ;DETERMINE THE CORRECT PAR'S
10261 055400 011003          MOV      (R0),R3          ;ADDRESS
10262 055402 005002          CLR      R2          ;GET ITS CONTENTS
    
```

```

10263 055404 073227 000006      ASHC   #6,R2      ;MAKE THE BLOCK COUNT
10264                               ;A 22-BIT ADDRESS.
10265 055410 060103      ADD    R1,R3      ;ADD THE OFFSET TO THE
10266 055412 005502      ADC    R2         ;BASE ADDRESS
10267
10268 055414 010237 055316      MOV    R2,TVADHI
10269 055420 010337 055314      MOV    R3,TVADLO
10270 055424 012746 055314      2$:   MOV    #TVADLO,-(SP) ;CALL $DB20 TO CONVERT THE
10271 055430 004737 053730      JSR    PC,$DB20   ;22-BIT
10272 055434 062716 000003      ADD    #3,(SP)   ;TYPE ONLY 8 DIGITS.
10273 055440 012637 055446      MOV    (SP)+,3$
10274 055444 104400
10275 055446 000000      3$:   .WORD  0
10276 055450 104414      RESREG
10277 055452 012616      MOV    (SP)+,(SP) ;RESTORE THE REGISTERS
10278                               ;LEAVE ONLY THE RETURN
10279 055454 000207      RTS    PC        ;ADDRESS ON THE STACK.
10280                               ;RETURN
10281                               .SBTTL SYSTEM DEVICE SIZER
10282
10283 ;THIS ROUTINE IS CALLED TO DETERMINE WHAT
10284 ;CONTROLLERS AND WHAT DRIVES ARE AVAILABLE ON
10285 ;THE SYSTEM.
10286 ;IT USES THE FLAGS:
10287 ;
10288 ;
10289 ;
10290 ;
10291 ;
10292 ;
10293 ;WHICH ARE BYTES CONTAINING A BIT FOR EACH
10294 ;POSSIBLE DEVICE ON THE CONTROLLER
10294 055456 005037 057372      SIZDEV: CLR   RS4FLG ;INITIALIZE FLAGE
10295 055462 005037 056422      CLR   RP4FLG
10296 055466 005037 062074      CLR   RH4FLG
10297 055472 005037 060326      CLR   RK5FLG
10298 055476 005037 061342      CLR   UBFLG
10299 055502 005037 057374      CLR   RS4ER1
10300 055506 005037 056424      CLR   RP4ER1
10301 055512 005037 062076      CLR   RH4ER1
10302 055516 005037 060330      CLR   RK5ER1
10303 055522 005037 061344      CLR   UBER1
10304 055526 104412      SAVREG
10305 055530 105037 056050      CLRB  RS4DFL
10306 055534 105037 056051      CLRB  RP4DFL
10307 055540 105037 056052      CLRB  RH4DFL
10308 055544 105037 056053      CLRB  RK5DFL
10309 055550 105037 056054      CLRB  UBDFL
10310
10311 055554 013737 000004 056056      MOV    @#4,SIZTM1 ;SAVE 4
10312 055562 012737 055610 000004      MOV    #1,@#4    ;IN CASE NON-EXISTENT REG.
10313
10314 055570 005777 126074      TST   @RS4CS1   ;TEST FOR RS04
10315 055574 004737 056070      OR    PC,SETREG ;GET THE REG ADD
10316 055600 003666      .WORD  RS4REG
10317 055602 004737 056130      JSR   PC,SIZRS4 ;GET THE # OF DRIVES
10318 055606 000403      BR    2$
    
```

```

10319
10320 055610 022626          1$:  CMP      (SP)+,(SP)+      ;THERE WAS NO RS04
10321 055612 005037 177766    CLR      @#CPUERR
10322
10323 055616 012737 055644 000004 2$:  MOV      #3$,@#4          ;IN CASE NON-EXISTENT REG.
10324
10325 055624 005777 126076    TST      @RP4CS1         ;TEST FOR RP04
10326 055630 004737 056070    JSR      PC,SETREG       ;GET THE REG ADD
10327 055634 003724          .WORD    RP4REG
10328 055636 004737 056214    JSR      PC,SIZRP4       ;GET THE # OF DRIVES
10329 055642 000403          BR       6$
10330
10331 055644 022626          3$:  CMP      (SP)+,(SP)+      ;THERE WAS NO RP04
10332 055646 005037 177766    CLR      @#CPUERR
10333
10334 055652 012737 055740 000004 6$:  MOV      #7$,@#4          ;IN CASE NON-EXISTENT REG.
10335 055660 005777 126120    TST      @RH4CS1         ;TEST FOR MASS BUS TESTER
10336 055664 004737 056070    JSR      PC,SETREG       ;GET THE REG ADD
10337 055670 004002          .WORD    RH4REG
10338 055672 012777 000007 126114  MOV      #7,@RH4CS2      ;SET THE DRIVE #
10339 055700 022777 000040 126124  CMP      #40,@RH4DT
10340 055706 001017          BNE      8$
10341 055710 013737 004004 004036  MOV      RH4CS1,RH4AE
10342 055716 062737 000074 004036  ADD      #74,RH4AE
10343 055724 004737 056070    JSR      PC,SETREG
10344 055730 004034          .WORD    RH4REX
10345
10346 055732 004737 056412    JSR      PC,SIZRH4
10347 055736 000403          BR       8$
10348
10349 055740 022626          7$:  CMP      (SP)+,(SP)+      ;THEREE WAS NO MBT
10350 055742 005037 177766    CLR      @#CPUERR
10351
10352 055746 012737 055774 000004 8$:  MOV      #9$,@#4          ;IN CASE NON-EXISTENT REG.
10353 055754 005777 126064    TST      @RK5DS          ;TEST FOR RK05
10354 055760 004737 056070    JSR      PC,SETREG       ;GET THE REG ADD
10355 055764 004042          .WORD    RK5REG
10356 055766 004737 056300    JSR      PC,SIZRK5       ;GET THE # OF DRIVES
10357 055772 000403          BR       10$
10358
10359 055774 022626          9$:  CMP      (SP)+,(SP)+      ;THERE WAS NO RK05
10360 055776 005037 177766    CLR      @#CPUERR
10361
10362 056002 012737 056030 000004 10$: MOV      #11$,@#4        ;IN CASE NON-EXISTENT REG
10363 056010 005777 126050    TST      @UBEDB         ;TEST FOR UNIBUS EXERCISER
10364 056014 004737 056070    JSR      PC,SETREG       ;GET THE REG ADD
10365 056020 004062          .WORD    UBereg
10366 056022 004737 056364    JSR      PC,SIZUBE       ;GET THE #
10367 056026 000403          BR       12$
10368
10369 056030 022626          11$: CMP      (SP)+,(SP)+      ;THERE WAS NO UBE
10370 056032 005037 177766    CLR      @#CPUERR
10371
10372 056036 013737 056056 000004 12$: MOV      SIZTM1,@#4      ;RESTORE 4
10373 056044 104414          RESREG
10374 056046 000207          RTS      PC
    
```

```

10375
10376 056050 000
10377 056051 000
10378 056052 000
10379 056053 000
10380 056054 000
10381 056056
10382
10383 056056 000000
10384 056060 000000
10385 056062 000000
10386 056064 000000
10387 056066 000000
10388
10389
10390
10391
10392
10393
10394
10395
10396
10397 056070 011637 056126
10398 056074 062716 000002
10399 056100 104412
10400 056102 017700 000020
10401 056106 012001
10402 056110 011002
10403 056112 010220
10404 056114 062702 000002
10405 056120 077104
10406 056122 104414
10407 056124 000207
10408
10409 056126 000000
10410
10411
10412
10413
10414
10415
10416 056130 012700 000010
10417 056134 012701 000001
10418 056140 005002
10419 056142 105037 056151
10420
10421 056146 104442
10422 056150 001
10423 056151 000
10424 056152 000000
10425 056154 000000
10426 056156 000000
10427 056160 000000
10428 056162 000000
10429 056164 000000
10430

```

```

RS4DFL: .BYTE 0
RP4DFL: .BYTE 0
RH4DFL: .BYTE 0
RK5DFL: .BYTE 0
UBEDFL: .BYTE 0
        .EVEN

```

```

SIZTM1: .WORD 0
SIZTM2: .WORD 0
SIZTM3: .WORD 0
SIZTM4: .WORD 0
SIZTM5: .WORD 0

```

```

;THIS ROUTINE IS CALLED BY A:
;
;      JSR      PC,SETREG
;      .WORD   DEVREG
;WHERE DEVREG IS THE STARTING ADDRESS OF
;A TABLE, WHICH IS TO CONTAIN THE ADDRESS OF
;A DEVICE'S CONTROL AND STATUS REGISTERS.
;THE TABLES ARE GENERATE HERE

```

```

SETREG: MOV      (SP),SETMP
        ADD      #2,(SP)
        SAVREG =
        MOV      1, @SETMP,R0
        MOV      (R0),R1
        MOV      (R0),R2
1$:     MOV      R2,(R0)+
        ADD      #2,R2
        SOB     R1,1$
        RESREG
        RTS     PC

```

```

SETMP: .WORD 0

```

```

;THIS ROUTINE IS CALLED, AFTER IT HAS BEEN
;DETERMINED IF THERE IS A RS04 CONTROLLER, TO SFE
;WHT DRIVES ARE AVAILABLE.

```

```

SIZRS4: MOV      #10,R0
        MOV      #1,R1
        CLR     R2
        CLRB   3$

```

```

1$:     \CALRS4
2$:     .BYTE 1
3$:     .BYTE 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0
        .WORD 0

```

```

;DO A NOP FUNCTION
;FOR EACH OF POSSIBLY
;8 DRIVES

```

10431 056166 005737 057374
 10432 056172 001001
 10433 056174 050102
 10434 056176 006301
 10435 056200 105237 056151
 10436 056204 077020
 10437
 10438 056206 110237 056050
 10439 056212 000207
 10440
 10441
 10442
 10443
 10444 056214 012700 000010
 10445 056220 012701 000001
 10446 056224 005002
 10447 056226 105037 056235
 10448
 10449 056232 104444
 10450 056234 021
 10451 056235 000
 10452 056236 000000
 10453 056240 000000
 10454 056242 000000
 10455 056244 000000
 10456 056246 000000
 10457 056250 000000
 10458
 10459 056252 005737 056424
 10460 056256 001001
 10461 056260 050102
 10462 056262 006301
 10463 056264 105237 056235
 10464 056270 077020
 10465
 10466 056272 110237 056051
 10467 056276 000207
 10468
 10469
 10470
 10471 056300 012700 000010
 10472 056304 012701 000001
 10473 056310 005002

TST RS4ER1
 BNE 4\$
 BIS R1,R2
 4\$: ASL R1
 INCB 3\$
 SOB R0,1\$
 MOVB R2,RS4DFL
 RTS PC
 ;THIS ROUTINE IS CALLED TO DETERMINE WHAT RP04
 ;DRIVES ARE ON THE CONTROLLER
 SIZRP4: MOV #10,R0
 MOV #1,R1
 CLR R2
 CLRB 3\$
 1\$: CALRP4 ;DO A READ IN PRESET
 2\$: .BYTE 21 ;FOR EACH OF UP TO
 3\$: .BYTE 0 ;8 DRIVES.
 .WORD 0
 .WORD 0
 .WORD 0
 .WORD 0
 .WORD 0
 .WORD 0
 TST RP4ER1
 BNE 4\$
 BIS R1,R2
 4\$: ASL R1
 INCB 3\$
 SOB R0,1\$
 MOVB R2,RP4DFL
 RTS PC
 ;DETERMINE WHAT RK05 DRIVES ARE AVAILABLE.
 SIZRK5: MOV #10,R0
 MOV #1,R1
 CLR R2

10474 056312 105037 056321 CLR 3\$
10475


```

10476 056316 104450 1$: CALRK5 ;DO A DRIVE RESET
10477 056320 015 .BYTE 15 ;FOR EACH OF 8
10478 056321 000 3$: .BYTE 0 ;POSSIBLE DRIVES.
10479 056322 000000 .WORD 0
10480 056324 000000 .WORD 0
10481 056326 000000 .WORD 0
10482 056330 000000 .WORD 0
10483 056332 000000 .WORD 0
10484 056334 000000 .WORD 0
10485
10486 056336 005737 060330 TST RK5R1
10487 056342 001001 BNE 4$
10488 056344 050102 BIS R1,R2
10489 056346 006301 4$: ASL R1
10490 056350 105237 056321 INCB 3$
10491 056354 077020 SOB R0,1$
10492
10493 056356 110237 056053 MOVB R2,RK5DFL
10494 056362 000207 RTS PC
10495
10496 ;SET UP UBEDFL
10497
10498 056364 042777 000200 125500 SIZUBE: BIC #BIT7,@UBECR1
10499 056372 032777 000200 125472 BIT #BIT7,@UBECR1
10500 056400 001403 BEQ 1$
10501 056402 112737 000001 056054 MOVB #1,UBEDFL
10502 056410 000207 1$: RTS PC
10503
10504 ;DETERMINE WHAT MASS BUS TESTER UNITS THERE ARE
10505
10506 056412 012737 000200 056052 SIZRH4: MOV #BIT7,RH4DFL
10507 056420 000207 RTS PC
10508
10509 .SBTTL DEVICE HANDLERS
10510 :*****
10511 :*
10512 :* THE FOLLOWING SIX ROUTINES:
10513 :* RH4HAN
10514 :* RP4HAN
10515 :* RS4HAN
10516 :* UBEHAN
10517 :* RK5HAN
10518 :* ARE O/I AND BUS TESTER DEVICE HANDLERS.
10519 :* THEY ARE CALLED USING:
10520 :* TRAP TABLE CALL
10521 :* FUNCTION: .BYTE
10522 :* UNITNUM: .BYTE
10523 :* DISKADR1: .WORD
10524 :* DISKADR2: .WORD
10525 :* MEMADR1: .WORD
10526 :* MEMADR2: .WORD
10527 :* WORDCNT: .WORD
10528 :* VECTOR: .WORD
10529 :* RETURN:
10530 :*A WHERE TRAP TABLE CALL IS ONE OF
10531 :* CALRH4
    
```

```

10532      : *          CALRP4
10533      : *          CALRS4
10534      : *          CALUBE
10535      : *          CALRK5
10536      : *B      FUNCTION IS THE PATTERN TO BE LOADED INTO THE
10537      : *          CONTROL REGISTER FUNCTION BITS, WITH EITHER
10538      : *          INTERRUPT ENABLED OR NOT.
10539      : *C      UNITNUM IS THE DRIVE NUMBER
10540      : *D      DISKADR1 AND DISKADR2 ARE THE DISK ADDRESS
10541      : *          SECTOR NUMBER
10542      : *E      MEMADR1 AND MEMADR2 ARE THE 22-BIT MEMORY
10543      : *          ADDRESS FOR THE TRANSFER.
10544      : *F      WORDCNT IS THE WORD COUNT A POSITIVE
10545      : *          NUMBER BETWEEN 0 AND 32K.
10546      : *G      VECTOR IS THE INTERRUPT HANDLER ROUTINE SPECIFIED
10547      : *          BY THE USER FOR AN INTERRUPT ENABLED FUNCTION.
10548      : *
10549      : *          WHEN THE HANDLER PROCESSES A CALL IT RETURNS
10550      : *          WITH THE FUNCTION IN PROGRESS IF THE
10551      : *          FUNCTION WAS INTERRUPT ENABLED. WHEN THE
10552      : *          INTERRUPT OCCURS CONTROL IS GIVEN TO
10553      : *          THE USER SPECIFIED INTERRUPT HANDLER.
10554      : *          IF THE FUNCTION WAS NOT INTERRUPT
10555      : *          ENABLED THEN THE HANDLER WAITS FOR
10556      : *          FUNCTION DONE BEFORE RETURNING.
10557      : *
10558      : *          THE FLAGS:
10559      : *          XXXER1
10560      : *          XXXER2
10561      : *          XXXER3
10562      : *          WHERE XXX IS THE DEVICE, ARE USED TO
10563      : *          INDICATE AND LOG DEVICE ERRORS IN THE HANDLER.
10564      : *          XXX CAN BE RH4,RP4,RS4,UBE,RK5 OR RP3.
10565      : *          XXXER1=0      NO ERRORS
10566      : *          XXXER1=1      ERRORS WITH STATUS IN XXXER2 AND XXXER3.
10567      : *
10568      : *****
10569
10570      :          .SBTTL          RP04 DISK HANDLER
10571      :RP04 DISK HANDLER
10572
10573      :REGISTERS USED IN RP4HAN
10574 056422 000000 RP4FLG: .WORD 0
10575 056424 000000 RP4ER1: .WORD 0          ;ERROR FLAGS.
10576 056426 000000 RP4ER2: .WORD 0
10577 056430 000000 RP4ER3: .WORD 0
10578 056432 000000 RP4ER4: .WORD 0
10579 056434 000000 RP4USE: .WORD 0
10580 056436 000000 RP4TMP: .WORD 0
10581 056440 000000 RP4FUN: .WORD 0
10582 056442 000000 RP4UNI: .WORD 0
10583 056444 000000 RP4DA1: .WORD 0
10584 056446 000000 RP4DA2: .WORD 0
10585 056450 000000 RP4MA1: .WORD 0
10586 056452 000000 RP4MA2: .WORD 0
10587 056454 000000 RP4WCT: .WORD 0
    
```

```

10588 056456 000000          RP4VEC:.WORD 0
10589 056460 000000          RP4TRK:.WORD 0
10590 056462 000000          RP4SEC:.WORD 0
10591 056464 000000          RP4CYL:.WORD 0
10592
10593 056466 005737 056422  RP4HAN: TST  RP4FLG      ;SEE IF THERE IS
10594 056472 001402          BEQ  RP4H1      ;ALREADY AN RP04 FUNCTION
10595 056474 104000          ERROR          ;IN PROGRESS. IF THERE
10596 056476 000000          HALT          ;IS ERROR> (SHOULD NEVER
10597 056500 012737 000340 177776 RP4H1: MOV  #340,@#PSW ;HAPPEN.)
10598 056506 011637 056436  MOV  (SP),RP4TMP ;RAISE THE PRIORITY
10599 056512 062716 000016  ADD  #16,(SP)
10600 056516 104412          SAVREG
10601 056520 013700 056436  MOV  RP4TMP,R0   ;GET AN ARGUMENT POINTER
10602 056524 112037 056440  MOVB (R0)+,RP4FUN ;RESET THE RETURN ADDRFS
10603 056530 112037 056442  MOVB (R0)+,RP4UNI ;FUNCTION
10604 056534 012037 056444  MOV  (R0)+,RP4DA1 ;UNIT, DEVICE, NUMBER
10605 056540 012037 056446  MOV  (R0)+,RP4DA2 ;DISK ADDRESS
10606 056544 012037 056450  MOV  (R0)+,RP4MA1 ;MEMORY ADDRESS
10607 056550 012037 056452  MOV  (R0)+,RP4MA2
10608 056554 012037 056454  MOV  (R0)+,RP4WCT ;WORD COUNT
10609 056560 012037 056456  MOV  (R0)+,RP4VEC ;INTERRUPT HANDLER ROUTINE
10610 056564 005037 056424  CLR  RP4ER1      ;CLEAR THE ERROR
10611 056570 005037 056426  CLR  RP4ER2      ;FLAGS
10612 056574 005037 056430  CLR  RP4ER3
10613
10614 056600 004737 057062  JSR  PC,RP4S1    ;GO SET UP THE UNIT NUMBER
10615 056604 004737 057132  JSR  PC,RP4RDY   ;GET THE DEVICE READY.
10616 056610 004737 057072  JSR  PC,RP4S2    ;COMPUTE THE CYLINDER,
10617                                     ;TRACK AND SECTOR
10618 056614 004737 057116  JSR  PC,RP4S3    ;SET UP THE WORD COUNT
10619
10620 056620 013777 056442 125110 RP4H2: MOV  RP4UNI,@RP4CS2 ;SET THE RP04 REGISTERS
10621 056626 013777 056454 125074  MOV  RP4WCT,@RP4WC  ;UP FOR THIS FUNCTION
10622 056634 013777 056450 125070  MOV  RP4MA1,@RP4BA
10623 056642 013777 056452 125126  MOV  RP4MA2,@RP4BAE
10624 056650 013777 056446 125056  MOV  RP4DA2,@RP4DA
10625 056656 013777 056444 125076  MOV  RP4DA1,@RP4DC
10626 056664 013700 004102  MOV  RP4V,R0      ;SET UP THE INTERRUPT
10627 056670 012720 056742  MOV  #RP4H4,(R0)+ ;VECTOR
10628 056674 012710 000340  MOV  #340,(R0)
10629 056700 013700 056440  MOV  RP4FUN,R0    ;LOAD THE FUNCTION
10630 056704 010037 056422  MOV  R0,RP4FLG    ;AND GO
10631 056710 110077 125012  MOVB R0,@RP4CS1
10632 056714 032700 000100  BIT  #BIT6,R0     ;SEE IF THE FUNCTION
10633 056720 001402          BEQ  RP4H3      ;WILL INTERRUPT WHEN
10634 056722 104414          RESREG          ;DONE. IF YES RETURN
10635 056724 000002          RTI           ;IF NOT INTERRUPTING
10636 056726 004737 056756  RP4H3: JSR  PC,RP4H5    ;THEN WAIT FOR THE
10637 056732 005037 056422  CLR  RP4FLG      ;FUNCTION TO FINISH.
10638 056736 104414          RESREG          ;THEN RETURN.
10639 056740 000002          RTI
10640
10641 056742 005037 056422  RP4H4: CLR  RP4FLG    ;WHEN THE INTERRUPT
10642 056746 004737 056756  JSR  PC,RP4H5    ;OCCURS CHECK FOR ERRORS
10643 056752 000177 177500  JMP  @RP4VEC     ;AND GO TO THE SERVICE
    
```

```

10644                                     ;ROUTINE.
10645
10646 056756 010046 RP4H5: MOV R0,-(SP)
10647 056760 053777 056442 124750 RP4H51: BIS RP4UNI,@RP4CS2
10648 056766 017700 124734 MOV @RP4CS1,R0
10649 056772 005700 TST R0 ;SEE IF THE FUNCTION
10650 056774 100023 BPL RP4H6 ;WAS COMPLETED WITHOUT
10651 056776 032700 060000 BIT #60000,R0 ;ERRORS.
10652 057002 001420 BEQ RP4H6
10653 057004 017737 124726 056426 MOV @RP4CS2,RP4ER2 ;IF ERRORS OCCURRED SET
10654 057012 017737 124722 056430 MOV @RP4DS,RP4ER3 ;THE INDICATORS
10655 057020 017737 124716 056432 MOV @RP4RR1,RP4ER4
10656 057026 012737 177777 056424 MOV #-1,RP4ER1
10657 057034 004737 057354 JSR PC,RP4CLR ;CLEAR THE CONTROL
10658 057040 012600 MOV (SP)+,R0
10659 057042 000207 RTS PC
10660 057044 105700 RP4H6: TSTB R0 ;WAIT FOR READY OR
10661 057046 100344 BPL RP4H51 ;ERROR
10662 057050 105777 124664 TSTB @RP4DS
10663 057054 100341 BPL RP4H51
10664 057056 012600 MOV (SP)+,R0
10665 057060 000207 RTS PC
10666
10667 057062 042737 177770 056442 RP4S1: BIC #177770,RP4UNI ;SET UP THE DRIVE NUMBER.
10668 057070 000207 RTS PC
10669
10670 057072 013701 056444 RP4S2: MOV RP4DA1,R1 ;COMPUTE THE DISK
10671 057076 005000 CLR R0
10672 057100 071027 000630 DIV #408.,R0
10673 057104 010137 056444 MOV R1,RP4DA1
10674 057110 005037 056446 CLR RP4DA2
10675 057114 000207 RTS PC
10676
10677 057116 005437 056454 RP4S3: NEG RP4WCT ;COMPUTE VALID WORD COUNT
10678 057122 042737 177700 056452 BIC #177700,RP4MA2 ;AND MEMORY ADDRESS
10679 057130 000207 RTS PC
10680
10681 057132 012737 000040 056434 RP4RDY: MOV #BIT5,RP4USE ;CLEAR CONTROLLER AND
10682 057140 053737 056442 056434 BIS RP4UNI,RP4USE
10683 057146 013777 056434 124562 MOV RP4USE,@RP4CS2
10684 057154 013777 056442 124554 MOV RP4UNI,@RP4CS2
10685 057162 105777 124540 1$: TSTB @RP4CS1 ;DRIVES
10686 057166 100375 BPL 1$
10687 057170 013777 056442 124540 MOV RP4UNI,@RP4CS2
10688 057176 012777 000021 124522 MOV #21,@RP4CS1 ;INITIALIZE THE DRIVE
10689 057204 017701 124516 2$: MOV @RP4CS1,R1 ;BY DOING A NOP
10690 057210 005701 TST R1 ;WAIT FOR ERROR OR
10691 057212 100434 BMI 4$ ;READY
10692 057214 105701 TSTB R1
10693 057216 100372 BPL 2$
10694
10695 057220 017700 124514 3$: MOV @RP4DS,R0 ;LOOK AT THE DRIVE
10696 ;STATUS
10697 057224 032700 000400 BIT #BIT8,R0 ;DRIVE PRESENT?
10698 057230 001430 BEQ 5$
10699 057232 032700 000100 BIT #BIT6,R0 ;VOLUME VALID?
    
```

```

10700 057236 001425      BEQ      5$
10701 057240 032700 010000  BIT      #BIT12,R0      ;ON LINE?
10702 057244 001422      BEQ      5$
10703 057246 032700 040000  BIT      #BIT14,R0      ;ANY ERRORS?
10704 057252 001017      BNE      5$
10705 057254 032700 004000  BIT      #BIT11,R0      ;WRITE LOCKED
10706 057260 001014      BNE      5$
10707 057262 032700 001000  BIT      #BIT9, R0      ;PROGRAMMABLE DRIVE
10708 057266 001011      BNE      5$
10709 057270 105700      TSTB    R0              ;WAIT FOR DRIVE READY
10710 057272 100344      BPL      2$
10711
10712 057274 012777 010000 124456  MOV      #BIT12,@RP40F    ;SET 16-BIT MODE
10713 057302 000207      RTS      PC              ;RETURN READY.
10714 057304 032701 040000      4$:    BIT      #BIT14,R1    ;ATTENTION OR ERROR?
10715 057310 001743      BEQ      3$
10716 057312 005726      5$:    TST      (SP)+
10717 057314 017737 124416 056426  MOV      @RP4CS2,RP4ER2  ;FLAG AND RECORD
10718 057322 017737 124412 056430  MOV      @RP4DS,RP4ER3  ;ERROR
10719 057330 017737 124406 056432  MOV      @RP4RR1,RP4ER4
10720 057336 012737 177777 056424  MOV      #-1,RP4ER1
10721 057344 004737 057354      JSR     PC,RP4CLR        ;CLR THE CONTROLLER
10722 057350 104414      RESREG
10723 057352 000002      RTI                    ;AND DRIVES.
10724                                     ;RETURN
10725 057354 013777 056434 124354  RP4CLR: MOV      RP4USE,@RP4CS2 ;CLR THE CONTROLLER
10726 057362 105777 124340      1$:    TSTB    @RP4CS1      ;AND DRIVES.
10727 057366 100375      BPL      1$
10728 057370 000207      RTS      PC
10729
10730                                     .SBTTL      RS04 DISK HANDLE
10731                                     ;RS04 DISK HANDLER
10732
10733                                     ;REGISTERS USED IN RS4HAN
10734 057372 000000      RS4FLG: .WORD 0
10735 057374 000000      RS4ER1: .WORD 0          ;ERROR FLAGS.
10736 057376 000000      RS4ER2: .WORD 0
10737 057400 000000      RS4ER3: .WORD 0
10738 057402 000000      RS4ER4: .WORD 0
10739 057404 000000      RS4USE: .WORD 0
10740 057406 000000      RS4TMP: .WORD 0
10741 057410 000000      RS4FUN: .WORD 0
10742 057412 000000      RS4UNI: .WORD 0
10743 057414 000000      RS4DA1: .WORD 0
10744 057416 000000      RS4DA2: .WORD 0
10745 057420 000000      RS4MA1: .WORD 0
10746 057422 000000      RS4MA2: .WORD 0
10747 057424 000000      RS4WCT: .WORD 0
10748 057426 000000      RS4VEC: .WORD 0
10749 057430 000000      RS4TRK: .WORD 0
10750 057432 000000      RS4SEC: .WORD 0
10751 057434 000000      RS4CYL: .WORD 0
10752
10753 057436 005737 057372      RS4HAN: TST      RS4FLG    ;SEE IF THERE ALREADY
10754 057442 001402      BEQ      RS4H1          ;IS AN RS04 FUNCTION
10755 057444 104000      ERROR      ;IN PROGRESS. IF SO
    
```

```

10756 057446 000000          HALT          ;ERROR. (SHOULD NEVER
10757 057450 012737 000340 177776 RS4H1: MOV      #340,@#PSW ;HAPPEN.
10758 057456 011637 057406          MOV      (SP),RS4TMP
10759 057462 062716 000016          ADD      #16,(SP)
10760 057466 104412          SAVREG
10761 057470 013700 057406          MOV      RS4TMP,R0 ;RAISE THE PRIORITY
10762 057474 112037 057410          MOV      (R0)+,RS4FUN ;GET A POINTER TO
10763 057500 112037 057412          MOV      (R0)+,RS4UNI ;FUNCTION
10764 057504 012037 057414          MOV      (R0)+,RS4DA1 ;GET THE DRIVE NUMBER
10765 057510 012037 057416          MOV      (R0)+,RS4DA2 ;DISK ADDRESS
10766 057514 012037 057420          MOV      (R0)+,RS4MA1 ;MEMORY ADDRESS
10767 057520 012037 057422          MOV      (R0)+,RS4MA2
10768 057524 012037 057424          MOV      (R0)+,RS4WCT ;WORD COUNT
10769 057530 012037 057426          MOV      (R0)+,RS4VEC ;INTERRUPT HANDLER ADDRESS
10770 057534 005037 057374          CLR      RS4ER1 ;CLEAR THE ERROR FLAGS
10771 057540 005037 057376          CLR      RS4ER2
10772 057544 005037 057400          CLR      RS4ER3
10773
10774 057550 004737 060024          JSR      PC,RS4S1 ;SET UP UNIT (DRIVE) NUMBER
10775 057554 004737 060116          JSR      PC,RS4RDY ;INITIALIZE DRIVE AND
10776                                     ;CONTROLLER
10777 057560 004737 060034          JSR      PC,RS4S2 ;COMPUTE TRACK AND SECTOR
10778 057564 004737 060102          JSR      PC,RS4S3 ;COMPUTE WORD COUNT.
10779
10780 057570 013777 057412 124102 RS4H2: MOV      RS4UNI,@RS4CS2 ;SET UP THE CONTROL
10781 057576 013777 057424 124066          MOV      RS4WCT,@RS4WC ;AND DRIVE REGISTERS
10782 057604 013777 057420 124062          MOV      RS4MA1,@RS4BA
10783 057612 013777 057422 124100          MOV      RS4MA2,@RS4BAE
10784 057620 013777 057414 124050          MOV      RS4DA1,@RS4DA
10785 057626 013700 004100          MOV      RS4V,R0
10786 057632 012720 057704          MOV      #RS4H4,(R0)+ ;SET THE INTERRUPT
10787 057636 012710 000340          MOV      #340,(R0)
10788 057642 013700 057410          MOV      RS4FUN,R0
10789 057646 010037 057372          MOV      R0,RS4FLG
10790 057652 110077 124012          MOV      R0,@RS4CS1 ;LOAD THE FUNCTION AND GO.
10791 057656 032700 000100          BIT      #BIT6,R0 ;SEE IF AN INTERRUPT
10792 057662 001402          BEQ      RS4H3 ;IS TO BE EXPECTED.
10793 057664 104414          RESREG ;IF YES THEN RETURN
10794 057666 000002          RTI
10795
10796 057670 004737 057720          RS4H3: JSR      PC,RS4H5 ;IF NOT INTERRUPTING
10797 057674 005037 057372          CLR      RS4FLG ;THEN WAIT FOR THE
10798 057700 104414          RESREG ;FUNCTION TO FINISH
10799 057702 000002          RTI
10800
10801 057704 005037 057372          RS4H4: CLR      RS4FLG ;WHEN THE INTERRUPT OCCURS.
10802 057710 004737 057720          JSR      PC,RS4H5 ;MAKE SURE THERE WERE
10803 057714 000177 177506          JMP      @RS4VEC ;NO ERRORS BEFORE GOING
10804                                     ;TO THE INTERRUPT
10805                                     ;SERVICE ROUTINE.
10806 057720 010046          RS4H5: MOV      R0,-(SP)
10807 057722 053777 057412 123750 RS4H51: BIS      RS4UNI,@RS4CS2
10808 057730 017700 123734          MOV      @RS4CS1,R0
10809 057734 005700          TST      R0 ;SEE IF THE FUNCTION
10810 057736 100023          BPL      RS4H6 ;WAS COMPLETED WITHOUT
10811 057740 032700 060000          BIT      #60000,R0 ;ERRORS
    
```

```

10812 057744 001420      BEQ      RS4H6
10813 057746 017737 123726 057376      MOV      @RS4CS2,RS4ER2 ;IF ERRORS OCCURRED
10814 057754 017737 123722 057400      MOV      @RS4DS,RS4ER3 ;SET THE INDICATORS
10815 057762 017737 123716 057402      MOV      @RS4ER,RS4ER4
10816 057770 012737 177777 057374      MOV      #-1,RS4ER1
10817 057776 004737 060310      JSR      PC,RS4CLR      ;THEN CLEAR THE CONTROL
10818 060002 012600      MOV      (SP)+,R0
10819 060004 000207      RTS      PC              ;AND DRIVES
10820 060006 105700      RS4H6: TSTB     R0
10821 060010 100344      BPL      RS4H51          ;WAIT FOR READY OR
10822 060012 105777 123664      TSTB     @RS4DS          ;ERROR
10823 060016 100341      BPL      RS4H51
10824 060020 012600      MOV      (SP)+,R0
10825 060022 000207      RTS      PC
10826
10827 060024 042737 177770 057412 RS4S1: BIC      #177770,RS4UNI ;SET UP DRIVE NUMBER
10828 060032 000207      RTS      PC
10829
10830 060034 013701 057414      RS4S2: MOV      RS4DA1,R1    ;COMPUTE A DISK
10831 060040 005000      CLR      R0              ;ADDRESS
10832 060042 071027 007000      DIV      #3584.,R0
10833 060046 005000      CLR      R0
10834 060050 071027 000100      DIV      #100,R0
10835 060054 010037 057430      MOV      R0,RS4TRK
10836 060060 010137 057434      MOV      R1,RS4CYL
10837 060064 000300      SWAB     R0
10838 060066 006200      ASR      R0
10839 060070 006200      ASR      R0
10840 060072 050001      BIS      R0,R1
10841 060074 010137 057414      MOV      R1,RS4DA1
10842 060100 000207      RTS      PC
10843
10844 060102 005437 057424      RS4S3: NEG      RS4WCT      ;COMPUTE A VALID WORD
10845 060106 042737 177700 057422      BIC      #177700,RS4MA2 ;COUNT AND MEMORY
10846 060114 000207      RTS      PC              ;ADDRESS
10847 060116 012737 000040 057404 RS4RDY: MOV      #BIT5,RS4USE ;CLEAR CONTROLLER AND DRIVES
10848 060124 053737 057412 057404      BIS      RS4UNI,RS4USE
10849 060132 013777 057404 123540      MOV      RS4USE,@RS4CS2
10850 060140 013777 057412 123532      MOV      RS4UNI,@RS4CS2
10851 060146 105777 123516      1$:      TSTB     @RS4CS1
10852 060152 100375      BPL      1$
10853 060154 013777 057412 123516      MOV      RS4UNI,@RS4CS2
10854 060162 012777 000001 123500      MOV      #1,@RS4CS1    ;INITIALIZE THE DRIVE
10855 060170 017701 123474      2$:      MOV      @RS4CS1,R1    ;BY DOING A NOP.
10856 060174 005701      TST      R1
10857 060176 100420      BMI      4$
10858 060200 105701      TSTB     R1
10859 060202 100372      BPL      2$
10860
10861 060204 017700 123472      3$:      MOV      @RS4DS,R0      ;LOOK AT THE DRIVE STATUS
10862 060210 032700 000400      BIT      #BIT8,R0      ;DRIVE PRESENT?
10863 060214 001414      BEQ      5$
10864 060216 032700 010000      BIT      #BIT12,R0     ;ON LINE?
10865 060222 001411      BEQ      5$
10866 060224 032700 004000      BIT      #BIT11,R0     ;WRITE LOCKED?
10867 060230 001006      BNE      5$
    
```

```

10868 060232 105700          TSTB    R0          ;DRIVE READY?
10869 060234 100355          BPL     2$
10870 060236 000207          RTS     PC
10871 060240 032701 040000      4$:    BIT     #BIT14,R1    ;ATTENTION OR ERROR?
10872 060244 001757          BEQ     3$
10873 060246 005726          5$:    TST     (SP)+
10874 060250 017737 123424 057376      MOV     @RS4CS2,RS4ER2    ;FLAG AND RECORD THE
10875 060256 017737 123420 057400      MOV     @RS4DS,RS4ER3    ;ERROR
10876 060264 017737 123414 057402      MOV     @RS4ER,RS4ER4
10877 060272 012737 177777 057374      MOV     #-1,RS4ER1
10878 060300 004737 060310      JSR     PC,RS4CLR        ;CLR THE CONTROLLER
10879 060304 104414          RESREG
10880 060306 000002          RTI

```

```

10881
10882 060310 013777 057404 123352  RS4CLR: MOV     RS4USE,@RS4CS1    ;CLR THE CONTROLLER
10883 060316 105777 123346      1$:    TSTB   @RS4CS1
10884 060322 100375          BPL     1$
10885 060324 000207          RTS     PC

```

```

10886
10887
10888          .SBTTL          RK05 DISK HANDLER
10889          ;RK05 DISK HANDLER

```

```

10890
10891          .REGISTERS USED IN RK5HAN
10892 060326 000000      RK5FLG: .WORD    0
10893 060330 000000      RK5ER1: .WORD    0          ;ERROR FLAGS.
10894 060332 000000      RK5ER2: .WORD    0
10895 060334 000000      RK5ER3: .WORD    0
10896 060336 000000      RK5ER4: .WORD    0
10897 060340 000000      RK5USE: .WORD    0
10898 060342 000000      RK5TMP: .WORD    0
10899 060344 000000      RK5FUN: .WORD    0
10900 060346 000000      RK5UNI: .WORD    0
10901 060350 000000      RK5DA1: .WORD    0
10902 060352 000000      RK5DA2: .WORD    0
10903 060354 000000      RK5MA1: .WORD    0
10904 060356 000000      RK5MA2: .WORD    0
10905 060360 000000      RK5WCT: .WORD    0
10906 060362 000000      RK5VEC: .WORD    0
10907 060364 000000      RK5TRK: .WORD    0
10908 060366 000000      RK5SEC: .WORD    0
10909 060370 000000      RK5CYL: .WORD    0

```

```

10910
10911 060372 005737 060326      RK5HAN: TST     RK5FLG        ;SEE IF THERE IS ALREADY AN
10912 060376 001402          BEQ     RK5H1              ;RK05 FUNCTION IN PROGRESS
10913 060400 104000          ERROR
10914 060402 000000          HALT

```

```

10915
10916 060404 012737 000340 177776  RK5H1:  MOV     #340,@#PSW        ;RAISE THE PRIORITY
10917 060412 011637 060342      MOV     (SP),RK5TMP
10918 060416 062716 000016      ADD     #16,(SP)
10919 060422 104412          SAVREG
10920 060424 013700 060342      MOV     RK5TMP,R0
10921 060430 112037 060344      MOVB   (R0)+,RK5FUN        ;GET THE ARGUMENTS.
10922 060434 112037 060346      MOVB   (R0)+,RK5UNI
10923 060440 012037 060350      MOV     (R0)+,RK5DA1

```



```

10924 060444 012037 060352      MOV      (R0)+,RK5DA2
10925 060450 012037 060354      MOV      (R0)+,RK5MA1
10926 060454 012037 060356      MOV      (R0)+,RK5MA2
10927 060460 012037 060360      MOV      (R0)+,RK5WCT
10928 060464 012037 060362      MOV      (R0)+,RK5VEC
10929
10930 060470 005037 060330      CLR      RK5ER1      ;CLR THE ERROR FLAGS
10931 060474 005037 060332      CLR      RK5ER2
10932 060500 005037 060334      CLR      RK5ER3
10933
10934 060504 004737 060754      JSR      PC,RK5S1      ;SET UP THE DRIVE NUMBER
10935 060510 004737 061160      JSR      PC,RK5RDY     ;GET THE DEVICE AND CONTROL
10936                                ;READY
10937 060514 004737 060776      JSR      PC,RK5S2     ;COMPUTE THE SURFACE
10938                                ;CYLINDER AND SECTOR
10939                                ;ADDRESS.
10940 060520 004737 061100      JSR      PC,RK5S3     ;SET UP A WORD COUNT,
10941                                ;THE UNIBUS MAP
10942                                ;AND BUS ADDRESS.
10943
10944 060524 005077 123320      RK5H2:  CLR      @RK5CS1
10945 060530 013777 060346      123320  MOV      RK5UNI,@RK5DA ;SET THE DEVICE REGISTERS
10946 060536 013777 060360      123306  MOV      RK5WCT,@RK5WC ;TO DO THE FUNCTION
10947 060544 013777 060354      123302  MOV      RK5MA1,@RK5BA
10948 060552 053777 060356      123270  BIS      RK5MA2,@RK5CS1
10949 060560 053777 060350      123270  BIS      RK5DA1,@RK5DA
10950 060566 013700 004106      MOV      RK5V,R0      ;LOAD THE INTERRUPT VECTOR
10951 060572 012720 060644      MOV      #RK5H4,(R0)+
10952 060576 012710 000340      MOV      #340,(R0)
10953 060602 013700 060344      MOV      RK5FUN,R0
10954 060606 010037 060326      MOV      R0,RK5FLG
10955 060612 050077 123232      BIS      R0,@RK5CS1   ;LOAD THE FUNCTION AND
10956                                ;GO
10957
10958 060616 032700 000100      BIT      #BIT6,R0     ;SEE IF THE FUNCTION WILL
10959 060622 001402                BEQ      RK5H3        ;INTERRUPT WHEN DONE.
10960 060624 104414                RESREG                ;IF YES RETURN
10961 060626 000002                RTI
10962
10963 060630 004737 060672      RK5H3:  JSR      PC,RK5H5   ;IF THE FUNCTION WAS
10964 060634 005037 060326      CLR      RK5FLG      ;NOT INTERRUPT ENABLED
10965 060640 104414                RESREG                ;WAIT FOR DONE OR ERROR.
10966 060642 000002                RTI
10967
10968 060644 004737 060672      RK5H4:  JSR      PC,RK5H5   ;SEE IF THERE WERE ANY ERRORS.
10969 060650 005037 060326      CLR      RK5FLG
10970 060654 012777 060670      123224  MOV      #1$,@RK5V
10971 060662 000230                SPL      0
10972 060664 000177 177472                JMP      @RK5VEC
10973 060670 000002      1$:    RTI
10974
10975 060672 010046                RK5H5:  MOV      R0,-(SP)
10976 060674 017700 123150      RK5H51: MOV      @RK5CS1,R0   ;SEE IF ANY ERROR OCCURRED
10977 060700 005700                TST      R0
10978 060702 100015                BPL      RK5H6
10979 060704 017737 123136 060332      MOV      @RK5ER,RK5ER2 ;IF YES, FLAG THE ERRGR
    
```

```

10980 060712 017737 123126 060334      MOV      @RK5DS,RK5ER3      ;AND SAVE THE STATUS
10981 060720 012737 177777 060330      MOV      #-1,RK5ER1
10982 060726 004737 061320              JSR      PC,RK5CLR
10983 060732 012600              MOV      (SP)+,R0
10984 060734 000207              RTS      PC
10985
10986 060736 105700              RK5H6:  TSTB   R0              ;WAIT FOR DONE OR
10987 060740 100355              BPL     RK5H51              ;ERROR
10988 060742 105777 123076      TSTB   @RK5DS
10989 060746 100352              BPL     RK5H51
10990 060750 012600              MOV      (SP)+,R0
10991 060752 000207              RTS      PC
10992
10993 060754 013700 060346      RK5S1:  MOV     RK5UNI,R0
10994 060760 072027 000015      ASH     #13,R0
10995 060764 042700 017777      BIC     #017777,R0
10996 060770 010037 060346      MOV     R0,RK5UNI
10997 060774 000207              RTS     PC
10998
10999 060776 013701 060350      RK5S2:  MOV     RK5DA1,R1          ;COMPUTE THE CYLINDER
11000 061002 005000              CLR     R0                  ;SURFACE AND SECTOR
11001 061004 071027 011100      DIV     #4672.,R0          ;DISK ADDRESS
11002 061010 005000              CLR     R0
11003 061012 071027 000030      DIV     #24.,R0
11004 061016 010002              MOV     R0,R2
11005 061020 005000              CLR     R0
11006 061022 071027 000014      DIV     #12.,R0
11007 061026 010237 060370      MOV     R2,RK5CYL
11008 061032 010137 060366      MOV     R1,RK5SEC
11009 061036 010037 060364      MOV     R0,RK5TRK
11010 061042 072227 000005      ASH     #5,R2
11011 061046 042702 160037      BIC     #160037,R2
11012 061052 072027 000004      ASH     #4,R0
11013 061056 042700 177757      BIC     #177757,R0
11014 061062 042701 177760      BIC     #177760,R1
11015 061066 050100              BJS     R1,R0
11016 061070 050200              BIS     R2,R0
11017 061072 010037 060350      MOV     R0,RK5DA1
11018 061076 000207              RTS     PC
11019
11020 061100 005437 060360      RK5S3:  NEG     RK5WCT          ;COMPUTE A VALID
11021                                ;WORD COUNT AND
11022                                ;SET THE UB MAP
11022 061104 013700 060354      MOV     RK5MA1,R0          ;REGISTERS
11023 061110 013701 060356      MOV     RK5MA2,R1
11024 061114 042701 177700      BIC     #177700,R1
11025 061120 012702 170300      MOV     #MAPL20,R2
11026 061124 012703 000010      MOV     #10,R3
11027 061130 010022              1$:   MOV     R0,(R2)+
11028 061132 010122              MOV     R1,(R2)+
11029 061134 062700 020000      ADD     #20000,R0
11030 061140 005501              ADC     R1
11031 061142 077306              SOB     R3,1$
11032 061144 012737 000040 060356      MOV     #40,RK5MA2
11033 061152 005037 060354      CLR     RK5MA1
11034 061156 000207              RTS     PC
11035
    
```

```

11036 061160 053777 060346 122670 RK5RDY: BIS      RK5UNI,@RK5DA      ;DO A CONTROL CLEAR
11037 061166 012777 000001 122654      MOV      #1,@RK5CS1      ;FUNCTION
11038 061174 105777 122650      1$:     TSTB     @RK5CS1
11039 061200 100375      BPL      1$
11040
11041 061202 053777 060346 122646      BIS      RK5UNI,@RK5DA      ;DO A DRIVE CLEAR
11042 061210 012777 000015 122632      MOV      #15,@RK5CS1     ;FUNCTION
11043
11044 061216 017701 122626      2$:     MOV      @RK5CS1,R1      ;WAIT FOR DONE OR
11045 061222 100420      BMI      5$              ;ERROR.
11046 061224 105701      TSTB     R1
11047 061226 100373      BPL      2$
11048
11049 061230 017701 122610      3$:     MOV      @RK5DS,R1
11050 061234 032701 000040      BIT      #BIT5,R1        ;WRITE ENABLED?
11051 061240 001011      BNE      5$
11052 061242 005777 122600      TST      @RK5ER
11053 061246 100406      BMI      5$
11054 061250 105701      TSTB     R1
11055 061252 100366      BPL      3$
11056 061254 032701 000100      BIT      #BIT6,R1
11057 061260 001763      BEQ      3$
11058 061262 000207      4$:     RTS      PC
11059
11060 061264 005726      5$:     TST      (SP)+
11061 061266 017737 122554 060332      MOV      @RK5ER,RK5ER2
11062 061274 017737 122544 060334      MOV      @RK5DS,RK5ER3
11063 061302 012737 177777 060330      MOV      #-1,RK5ER1
11064 061310 004737 061320      JSR      PC,RK5CLR
11065 061314 104414      RESREG
11066 061316 000002      RTI
11067
11068 061320 005077 122532      RK5CLR: CLR      @RK5DA      ;RESET THE CONTROLLER
11069 061324 012777 000001 122516      MOV      #1,@RK5CS1     ;BY DOING A CONTROL
11070 061332 105777 122512      1$:     TSTB     @RK5CS1     ;CLEAR FUNCTION
11071 061336 100375      BPL      1$
11072 061340 000207      RTS      PC
11073
11074      .SBTTL      UNIBUS EXERCISER HANDLER
11075      ;UNIBUS EXERCISER HANDLER
11076
11077      ;REGISTERS USED IN UBEHAN
11078 061342 000000      UBEFLG: .WORD      0
11079 061344 000000      UBEER1: .WORD      0      ;ERROR FLAGS.
11080 061346 000000      UBEER2: .WORD      0
11081 061350 000000      UBEER3: .WORD      0
11082 061352 000000      UBEER4: .WORD      0
11083 061354 000000      UBEUSE: .WORD      0
11084 061356 000000      UBETMP: .WORD      0
11085 061360 000000      UBEFUN: .WORD      0
11086 061362 000000      UBEUNI: .WORD      0
11087 061364 000000      UBEDA1: .WORD      0
11088 061366 000000      UBEDA2: .WORD      0
11089 061370 000000      UBEMA1: .WORD      0
11090 061372 000000      UBEMA2: .WORD      0
11091 061374 000000      UBEWCT: .WORD      0
    
```

```

11092 061376 000000      UBEVEC: .WORD      0
11093 061400 000000      UBETRK: .WORD      0
11094 061402 000000      UBESEC: .WORD      0
11095 061404 000000      UBECYL: .WORD      0
11096
11097 061406 005737 061342      UBEHAN: TST        UBEFLG      ;SEE IF THERE IS ALREADY
11098 061412 001402          BEQ          UBEH1          ;A UNIBUS EXERCISER FUNCTION
11099 061414 104000          ERROR
11100 061416 000000          HALT          ;IN PROGRESS. IF THERE
11101                                     ;IS ERROR. (SHOULD NEVER HAPPEN)
11102 061420 012737 000340 177776 UBEH1:  MOV        #340,@#PSW      ;RAISE THE PRIORITY
11103 061426 011637 061356          MOV        (SP),UBETMP      ;GET AN ARGUMENT POINTER
11104 061432 062716 000016          ADD        #16,(SP)
11105 061436 104412          SAVREG
11106 061440 013700 061356          MOV        UBETMP,R0        ;RESET THE RETURN ADDRESS
11107
11108 061444 012037 061360          MOV        (R0)+,UBEFUN     ;GET THE ARGUMENTS.
11109 061450 012037 061364          MOV        (R0)+,UBEDA1
11110 061454 012037 061366          MOV        (R0)+,UBEDA2
11111 061460 012037 061370          MOV        (R0)+,UBEMA1
11112 061464 012037 061372          MOV        (R0)+,UBEMA2
11113 061470 012037 061374          MOV        (R0)+,UBEWCT
11114 061474 012037 061376          MOV        (R0)+,UBEVEC
11115 061500 005037 061344          CLR        UBEER1          ;CLEAR THE ERROR FLAGS
11116 061504 005037 061346          CLR        UBEER2
11117 061510 005037 061350          CLR        UBEER3
11118 061514 004737 062004          JSR        PC,UBERDY
11119 061520 004737 061722          JSR        PC,UBES1        ;GO SET UP THE BUS
11120                                     ;ADDRESS AND UB MAP
11121
11122 061524 013777 061374 122334 UBEH2:  MOV        UBEWCT,@UBECC     ;SET THE DEVICE
11123 061532 013777 061370 122330          MOV        UBEMA1,@UBEBA    ;REGISTERS
11124 061540 053777 061372 122330          BIS        UBEMA2,@UBECR2
11125 061546 013777 061366 122310          MOV        UBEDA2,@UBEDB
11126 061554 013700 004110          MOV        UBEV,R0
11127 061560 012720 061632          MOV        #UBEH4,(R0)+
11128 061564 012710 000340          MOV        #340,(R0)
11129 061570 013700 061360          MOV        UBEFUN,R0
11130 061574 010037 061342          MOV        R0,UBEFLG
11131 061600 010077 122266          MOV        R0,@UBECR1      ;LOAD THE FUNCTION
11132 061604 032700 000100          BIT        #BIT6,R0        ;SEE IF THE FUNCTION
11133 061610 001402          BEQ          UBEH3        ;IS INTERRUPT ENABLED
11134 061612 104414          RESREG      ;IF YES RETURN
11135 061614 000002          RTI
11136
11137 061616 004737 061646          UBEH3:  JSR        PC,UBEH5    ;IF NOT INTERRUPT ENABLED
11138 061622 005037 061342          CLR        UBEFLG        ;WAIT FOR DONE OR
11139 061626 104414          RESREG      ;ERROR
11140 061630 000002          RTI
11141
11142 061632 005037 061342          UBEH4:  CLR        UBEFLG      ;WHEN THE INTERRUPT
11143 061636 004737 061646          JSR        PC,UBEH5        ;OCCURS SEE IF ANY ERRORS
11144 061642 000177 177530          JMP        @UBEVEC        ;OCCURRED
11145
11146 061646 010046          UBEH5:  MOV        R0,-(SP)
11147 061650 017700 122216          UBEH51: MOV        @UBECR1,R0 ;WAIT FOR DONE OR
    
```

```

11148 061654 005700          TST      R0          ;ERROR
11149 061656 100015          BPL      UBEH6
11150
11151 061660 017737 122206 061346      MOV      @UBECR1,UBEER2
11152 061666 017737 122204 061350      MOV      @UBECR2,UBEER3
11153 061674 012737 177777 061344      MOV      #-1,UBEER1
11154 061702 004737 062060          JSR      PC,UBCLR
11155 061706 012600          MOV      (SP)+,R0
11156 061710 000207          RTS      PC
11157
11158 061712 105700          UBEH6:  TSTB     R0
11159 061714 100355          BPL      UBEH51
11160 061716 012600          MOV      (SP)+,R0
11161 061720 000207          RTS      PC
11162
11163 061722 013700 061370          UBE51:  MOV      UBEMA1,R0      ;SET UP THE BUS ADDRESS
11164 061726 013701 061372          MOV      UBEMA2,R1      ;AND UB MAPPING BOX
11165 061732 042701 177700          BIC      #177700,R1
11166 061736 012702 170200          MOV      #MAPL00,R2
11167 061742 012703 000010          MOV      #10,R3
11168
11169 061746 010022          1$:    MOV      R0,(R2)+
11170 061750 010122          MOV      R1,(R2)+
11171 061752 062700 020000          ADD      #20000,R0
11172 061756 005501          ADC      R1
11173 061760 077306          SOB      R3,1$
11174
11175 061762 005037 061372          CLR      UBEMA2
11176 061766 005037 061370          CLR      UBEMA1
11177 061772 005137 061374          COM      UBEWCT
11178 061776 005237 061374          INC      UBEWCT
11179 062002 000207          RTS      PC
11180
11181 062004 005077 122064          UBERDY: CLR      @UBECLR      ;TRY TO GET DEVICE
11182                                     ;READY
11183 062010 017700 122056          1$:    MOV      @UBECR1,R0
11184 062014 100403          BMI      2$
11185 062016 105700          TSTB     R0
11186 062020 100373          BPL      1$
11187 062022 000207          RTS      PC
11188
11189 062024 005726          2$:    TST      (SP)+
11190 062026 017737 122040 061346      MOV      @UBECR1,UBEER2
11191 062034 017737 122036 061350      MOV      @UBECR2,UBEER3
11192 062042 012737 177777 061350      MOV      #-1,UBEER3
11193 062050 004737 062060          JSR      PC,UBCLR
11194 062054 104414          RESREG
11195 062056 000002          RTI
11196
11197 062060 005077 122010          UBCLR:  CLR      @UBECLR      ;CLEAR THE DEVICE.
11198 062064 105777 122002          1$:    TSTB     @UBECR1
11199 062070 100375          BPL      1$
11200 062072 000207          RTS      PC
11201
11202                                     .SBTTL      MASS BUS TESTER HANDLER
11203                                     ;THIS CODE IS FOR HANDLING THE MASS BUS
    
```

```

11204          ;TESTED DEVICE.
11205
11206          ;REGISTERS USED IN RH4HAN
11207 062074 000000 RH4FLG: .WORD 0
11208 062076 000000 RH4ER1: .WORD 0          ;ERROR FLAGS.
11209 062100 000000 RH4ER2: .WORD 0
11210 062102 000000 RH4ER3: .WORD 0
11211 062104 000000 RH4ER4: .WORD 0
11212 062106 000000 RH4USE: .WORD 0
11213 062110 000000 RH4TMP: .WORD 0
11214 062112 000000 RH4FUN: .WORD 0
11215 062114 000000 RH4UNI: .WORD 0
11216 062116 000000 RH4DA1: .WORD 0
11217 062120 000000 RH4DA2: .WORD 0
11218 062122 000000 RH4MA1: .WORD 0
11219 062124 000000 RH4MA2: .WORD 0
11220 062126 000000 RH4WCT: .WORD 0
11221 062130 000000 RH4VEC: .WORD 0
11222 062132 000000 RH4TRK: .WORD 0
11223 062134 000000 RH4SEC: .WORD 0
11224 062136 000000 RH4CYL: .WORD 0
11225
11226 062140 005737 062074 RH4HAN: TST RH4FLG          ;SEE IF A FUNCTION
11227 062144 001402          BEQ RH4H1          ;IS ALREADY ACTIVE IF
11228 062146 104000          ERROR          ;SO ERROR.
11229 062150 000000          HALT
11230
11231 062152 012777 000340 115616 RH4H1: MOV #340,@PSW          ;RAISE THE PRIORITY
11232 062160 011637 062110          MOV (SP),RH4TMP
11233 062164 062716 000016          ADD #16,(SP)
11234 062170 104412          SAVREG
11235 062172 013700 062110          MOV RH4TMP,R0          ;RESET THE RETURN
11236 062176 112037 062112          MOVB (R0)+,RH4FUN
11237 062202 112037 062114          MOVB (R0)+,RH4UNI
11238 062206 012037 062116          MOV (R0)+,RH4DA1
11239 062212 012037 062120          MOV (R0)+,RH4DA2
11240 062216 012037 062122          MOV (R0)+,RH4MA1
11241 062222 012037 062124          MOV (R0)+,RH4MA2
11242 062226 012037 062126          MOV (R0)+,RH4WCT
11243 062232 011037 062130          MOV (R0),RH4VEC
11244 062236 005037 062076          CLR RH4ER1          ;CLEAR THE ERROR FLAGS
11245 062242 005037 062100          CLR RH4ER2
11246 062246 005037 062102          CLR RH4ER3
11247 062252 004737 062532          JSR PC,RH4S1          ;SET UP THE UNIT NUMBER
11248 062256 004737 062556          JSR PC,RH4RDY          ;GET THE UNIT READY
11249 062262 004737 062542          JSR PC,RH4S2
11250
11251 062266 013777 062114 121520 RH4H2: MOV RH4UNI,@RH4CS2          ;SET THE CONTROL REGISTERS
11252 062274 013777 062126 121504          MOV RH4WCT,@RH4WC          ;AND DEVICE REGISTERS
11253 062302 013777 062122 121500          MOV RH4MA1,@RH4BA
11254 062310 013777 062124 121520          MOV RH4MA2,@RH4AE
11255 062316 013777 062116 121500          MOV RH4DA1,@RH4DR
11256 062324 012777 004000 121476          MOV #4000,@RH4MR1
11257 062332 000240          NOP
11258 062334 013700 004104          MOV RH4V,R0          ;VECTOR
11259 062340 012720 062412          MOV #RH4H4,(R0)+
    
```

```

11260 062344 012710 000340      MOV      #340,(R0)
11261 062350 013700 062112      MOV      RH4FUN,R0
11262 062354 010037 062074      MOV      R0,RH4FLG      ;LOAD THE FUNCTION AND
11263 062360 110077 121420      MOV      R0,@RH4CS1      ;GO
11264 062364 032700 000100      BIT      #BIT6,R0      ;SEE IF THIS FUNCTION
11265 062370 001402      BEQ      RH4H3      ;WILL INTERRUPT WHEN DONE
11266 062372 104414      RESREG
11267 062374 000002      RTI      ;IF YES RETURN TO CALL
11268
11269 062376 004737 062426      RH4H3: JSR      PC,RH4H5      ;IF NOT INTERRUPT
11270 062402 005037 062074      CLR      RH4FLG      ;ENABLED WAIT FOR
11271 062406 104414      RESREG      ;THE FUNCTION TO
11272 062410 000002      RTI      ;FINISH THEN RETURN.
11273
11274 062412 005037 062074      RH4H4: CLR      RH4FLG      ;WHEN THE INTERRUPT
11275 062416 004737 062426      JSR      PC,RH4H5      ;OCCURS CHECKS FOR
11276 062422 000177 177502      JMP      @RH4VEC      ;ERRORS. THEN GO TO THE
11277
11278
11279
11280 062426 010046      RH4H5: MOV      R0,-(SP)
11281 062430 053777 062114 121356 RH4H51: BIS      RH4UNI,@RH4CS2
11282 062436 017700 121342      MOV      @RH4CS1,R0      ;SEE IF THE FUNCTION
11283 062442 005700      TST      R0      ;WAS COMPLETED WITHOUT
11284 062444 100023      BPL      RH4H6      ;ERRORS.
11285 062446 032700 060000      BIT      #60000,R0
11286 062452 001420      BEQ      RH4H6
11287 062454 017737 121334 062100      MOV      @RH4CS2,RH4ER2      ;IF ERRORS OCCURRED
11288 062462 017737 121330 062102      MOV      @RH4ST,RH4ER3      ;SAVE STATUS AND SET
11289 062470 017737 121324 062104      MOV      @RH4ER,RH4ER4
11290 062476 012737 177777 062076      MOV      #-1,RH4ER1      ;ERROR FLAGS.
11291 062504 004737 062750      JSR      PC,RH4CLR
11292 062510 012600      MOV      (SP)+,R0
11293 062512 000207      RTS      PC
11294
11295 062514 105700      RH4H6: TSTB     R0      ;WAIT FOR READY OR
11296 062516 100344      BP_     RH4H51      ;ERROR
11297 062520 105777 121272      TSTB     @RH4ST
11298 062524 100341      BPL     RH4H51
11299 062526 012600      MOV     (SP)+,R0
11300 062530 000207      RTS     PC
11301
11302 062532 042737 177770 062114 RH4S1: BIC     #177770,RH4UNI ;SET UP THE DRIVE NUMBER
11303 062540 000207      RTS     PC
11304
11305 062542 012737 000000 062120 RH4S2: MOV     #0,RH4DA2      ;FOR DEBUG.
11306 062550 005437 062126      NEG     RH4WCT      ;SET UP WORD COUNT
11307 062554 000207      RTS     PC
11308
11309 062556 012737 000040 062106 RH4RDY: MOV     #BITS,RH4USE ;CLR THE CONTROLLER
11310 062564 053737 062114 062106      BIS     RH4UNI,RH4USE
11311 062572 013777 062106 121214      MOV     RH4USE,@RH4CS2
11312 062600 013777 062114 121206      MOV     RH4UNI,@RH4CS2
11313 062606 105777 121172      1$:    TSTB     @RH4CS1 ;AND DRIVES
11314 062612 100375      BPL     1$
11315 062614 013777 062114 121172      MOV     RH4UNI,@RH4CS2 ;DO A NOP FUNCTION
    
```

```

11316 062622 012777 000001 121154      MOV      #1,@RH4CS1      ;TO INITIALIZE THE
11317                                     ;DRIVE
11318 062630 017701 121150      2$:     MOV      @RH4CS1,R1      ;WAIT FOR READY OR ERROR.
11319 062637 005701                                     TST      R1
11320 062636 100420                                     BMI      4$
11321 062640 105701                                     TSTB    R1
11322 062642 100372                                     BPL     2$
11323
11324 062644 017700 121146      3$:     MOV      @RH4ST,R0      ;LOOK AT THE UNIT STATUS
11325 062650 032700 000400                                     BIT      #BIT8,R0      ;UNIT PRESENT?
11326 062654 001414                                     BEQ     5$
11327 062656 032700 010000                                     BIT      #BIT12,R0     ;ON LINE?
11328 062662 001411                                     BFO     5$
11329 062664 032700 040000                                     BIT      #BIT14,R0     ;ANY ERRORS?
11330 062670 001006                                     BNE     5$
11331 062672 105700                                     TSTB    R0      ;WAIT FOR UNIT READY
11332 062674 100355                                     BPL     2$
11333 062676 000207                                     RTS     PC
11334
11335 062700 032701 040000      4$:     BIT      #BIT14,R1      ;ATTENTION OR ERROR
11336 062704 001757                                     BEQ     3$
11337 062706 005726      5$:     TST      (SP)+      ;FLAG AND RECORD ERROR
11338 062710 017737 121100 062100      MOV      @RH4CS2,RH4ER2
11339 062716 017737 121074 062102      MOV      @RH4ST,RH4ER3
11340 062724 017737 121070 062104      MOV      @RH4ER,RH4ER4
11341 062732 012737 177777 062076      MOV      #-1,RH4ER1
11342 062740 004737 062750      JSR      PC,RH4CLR
11343 062744 104414      RESREG
11344 062746 000002      RTI
11345
11346 062750 013777 062106 121036  RH4CLR: MOV      RH4USE,@RH4CS2      ;CLR THE CONTROLLER
11347 062756 105777 121022      1$:     TSTB    @RH4CS1      ;AND DRIVES.
11348 062762 100375                                     BPL     1$
11349 062764 000207      TSTD1:  RTS     PC
11350 062766 001000      .BLKW   512.
11351      ;SPECIAL MESSAGES:
11352
11353 064766 041536 000200      LONCMS: .ASCIZ  '^C'<CRLF>
11354
11355 064772 047515 044516 047524  MMESRS: .ASCIZ  'MONITOR (OR LOADER) RESTORED!'<CRLF>
11356 065000 020122 047450 020122
11357 065006 047514 042101 051105
11358 065014 020051 042522 052123
11359 065022 051117 042105 100041
11360 065030      000
11361
11362 065031      200 047520 042527  POWERM: .ASCIZ  <CRLF>'POWER FAILURE, PROGRAM RESTARTING'<CRLF><CRLF>
11363 065036 020122 040506 046111
11364 065044 051125 026105 050040
11365 065052 047522 051107 046501
11366 065060 051040 051505 040524
11367 065066 052122 047111 100107
11368 065074 000200
11369
11370 065076 000011      $TAB:   .ASCIZ  <TAB>
11371
    
```


11372	065100	042600	050130	041505	MTA5:	.ASCII	<CRLF>'EXPECTED DATA:'<CRLF>
11373	065106	042524	020104	040504			
11374	065114	040524	100072				
11375	065120	051107	052517	020120		.ASCIZ	'GROUP 0.GROUP 1.MEM EV.' <tab>'mem odd.'<crlf><="" td=""></tab>'mem>
11376	065126	027060	051107	052517			
11377	065134	020120	027061	042515			
11378	065142	020115	053105	004456			
11379	065150	042515	020115	042117			
11380	065156	027104	000200				
11381							
11382	065162	042200	052101	020101	MTA11:	.ASCII	<CRLF>'DATA WRITTEN.' <tab>'test addr.'<tab>'error="" reg.'<crlf><="" td=""></tab>'test>
11383	065170	051127	052111	042524			
11384	065176	027116	052011	051505			
11385	065204	020124	042101	051104			
11386	065212	004456	051105	047522			
11387	065220	020122	042522	027107			
11388	065226	200					
11389							
11390	065227	040	047111	000040	MTA17:	.ASCIZ	' IN '
11391							
11392	065234	054105	042520	052103	MTB17:	.ASCIZ	'EXPECTED DATA:'<CRLF>
11393	065242	042105	042040	052101			
11394	065250	035101	000200				
11395							
11396	065254	054502	042524	004456	MTC17:	.ASCIZ	'BYTE.' <tab>< td=""></tab><>
11397	065262	000					
11398							
11399	065263	127	051117	027104	MTA20:	.ASCIZ	'WORD.' <tab>< td=""></tab><>
11400	065270	000011					
11401							
11402	065272	054105	042520	052103	MTA21:	.ASCII	'EXPECTED DATA:'<CRLF>
11403	065300	042105	042040	052101			
11404	065306	035101	200				
11405	065311	110	052111	020123		.ASCIZ	'HITS IN GROUP 0.' <tab>' '<tab>'hits="" '<crlf><="" 1.="" group="" in="" td=""></tab>'>
11406	065316	047111	043440	047522			
11407	065324	050125	030040	004456			
11408	065332	004457	044510	051524			
11409	065340	044440	020116	051107			
11410	065346	052517	020120	027061			
11411	065354	100040	000				
11412							
11413		065227			MTB21-MTA17		
11414							
11415	065357	200	042524	052123	MTA43:	.ASCII	<CRLF>'TEST ADDRESS.' <tab>'error adrs="" reg.'<tab><="" td=""></tab>'error>
11416	065364	040440	042104	042522			
11417	065372	051523	004456	051105			
11418	065400	047522	020122	042101			
11419	065406	051522	051040	043505			
11420	065414	004456					
11421	065416	051105	047522	020122		.ASCIZ	'ERROR REG.' <crlf>< td=""></crlf><>
11422	065424	042522	027107	000200			
11423							
11424	065432	053600	047522	042524	MTA45:	.ASCIZ	<CRLF>'WROTE. 377' <tab>'in '<="" byte.="" td=""></tab>'in>
11425	065440	020056	033463	004467			
11426	065446	047111	041040	052131			
11427	065454	027105	000040				

11428						
11429	065460	051200	040505	020104	MTB45:	.ASCIZ <CRLF>'READ DATA. '
11430	065466	040504	040524	020056		
11431	065474	000				
11432						
11433	065475	011	047111	053440	MTC45:	.ASCIZ <TAB>'IN WORD. '
11434	065502	051117	027104	000040		
11435						
11436	065510	053600	047522	042524	MTA50:	.ASCIZ <CRLF>'WROTE. 000'<TAB>'IN BYTE. '
11437	065516	020056	030060	004460		
11438	065524	047111	041040	052131		
11439	065532	027105	000040			
11440						
11441	065536	042600	052116	051105	PDMSG1:	.ASCII <CRLF>'ENTERING CACHE ADDRESS MEMORY POWER UP '
11442	065544	047111	020107	040503		
11443	065552	044103	020105	042101		
11444	065560	051104	051505	020123		
11445	065566	042515	047515	054522		
11446	065574	050040	053517	051105		
11447	065602	052440	020120			
11448	065606	047111	040526	044514		.ASCII 'INVALIDATOR TEST.'<CRLF>
11449	065614	040504	047524	020122		
11450	065622	042524	052123	100056		
11451	065630	046120	040505	042523		.ASCII 'PLEASE GO THROUGH A POWER DOWN, POWER UP '
11452	065636	043440	020117	044124		
11453	065644	047522	043525	020110		
11454	065652	020101	047520	042527		
11455	065660	020122	047504	047127		
11456	065666	020054	047520	042527		
11457	065674	020122	050125	040		
11458	065701	123	050505	042525		.ASCIZ 'SEQUENCE.'<CRLF>
11459	065706	041516	027105	000200		
11460						
11461	065714	041600	041501	042510	PDMSG2:	.ASCII <CRLF>'CACHE ADDRESS MEMORY POWER UP INVALIDATOR'
11462	065722	040440	042104	042522		
11463	065730	051523	046440	046505		
11464	065736	051117	020131	047520		
11465	065744	042527	020122	050125		
11466	065752	044440	053116	046101		
11467	065760	042111	052101	051117		
11468	065766	052040	051505	020124		.ASCIZ ' TEST DID NOT FAIL.'<CRLF>
11469	065774	044504	020104	047516		
11470	066002	020124	040506	046111		
11471	066010	100056	000			
11472						
11473	066013	105	051122	051117	ADRNG:	.ASCII 'ERROR ADDRESS REGISTER NEEDED FOR TEST,'<CRLF>'BUT IT HAS BEEN '
11474	066020	040440	042104	042522		
11475	066026	051523	051040	043505		
11476	066034	051511	042524	020122		
11477	066042	042516	042105	042105		
11478	066050	043040	051117	052040		
11479	066056	051505	026124	041200		
11480	066064	052125	044440	020124		
11481	066072	040510	020123	042502		
11482	066100	047105	040			
11483	066103	106	040514	043507		.ASCIZ 'FLAGGED AS BAD.'

11484	066110	042105	040440	020123	
11485	066116	040502	020504	000	
11486					
11487	066123	105	051122	051117	ERRNG: .ASCII 'ERROR REGISTER NEEDED FOR TEST,'<CRLF>'BUT IT HAS BEEN '
11488	066130	051040	043505	051511	
11489	066136	042524	020122	042516	
11490	066144	042105	042105	043040	
11491	066152	051117	052040	051505	
11492	066160	026124	041200	052125	
11493	066166	044440	020124	040510	
11494	066174	020123	042502	047105	
11495	066202	040			
11496	066203	106	040514	043507	.ASCIZ 'FLAGGED AS BAD.'
11497	066210	042105	040440	020123	
11498	066216	040502	020504	000	
11499					
11500	066223	103	047117	051124	CNRNG: .ASCII 'CONTROL REGISTER NEEDED FOR TEST,'<CRLF>'BUT IT HAS BEEN '
11501	066230	046117	051040	043505	
11502	066236	051511	042524	020122	
11503	066244	042516	042105	042105	
11504	066252	043040	051117	052040	
11505	066260	051505	026124	041200	
11506	066266	052125	044440	020124	
11507	066274	040510	020123	042502	
11508	066302	047105	040		
11509	066305	106	040514	043507	.ASCIZ 'FLAGGED AS BAD '
11510	066312	042105	040440	020123	
11511	066320	040502	020504	000	
11512	066325	115	044501	052116	MNRNG: .ASCII 'MAINTENANCE REGISTER NEEDED FOR TEST,'<CRLF>'BUT IT HAS BEEN '
11513	066332	047105	047101	042503	
11514	066340	051040	043505	051511	
11515	066346	042524	020122	042516	
11516	066354	042105	042105	043040	
11517	066362	051117	052040	051505	
11518	066370	026124	041200	052125	
11519	066376	044440	020124	040510	
11520	066404	020123	042502	047105	
11521	066412	040			
11522	066413	106	040514	043507	.ASCIZ 'FLAGGED AS BAD.'
11523	066420	042105	040440	020123	
11524	066426	040502	020504	000	
11525					
11526	066433	110	052111	046457	HMRNG: .ASCII 'HIT/MISS REGISTER NEEDED FOR TEST,'<CRLF>'BUT IT HAS BEEN '
11527	066440	051511	020123	042522	
11528	066446	044507	052123	051105	
11529	066454	047040	042505	042504	
11530	066462	020104	047506	020122	
11531	066470	042524	052123	100054	
11532	066476	052502	020124	052111	
11533	066504	044040	051501	041040	
11534	066512	042505	020116		
11535	066516	046106	043501	042507	.ASCIZ 'FLAGGED AS BAD.'
11536	066524	020104	051501	041040	
11537	066532	042101	000041		
11538					
11539	066536	040600	042104	042522	MTA77: .ASCIZ <CRLF>'ADDRESS: '

11540	066544	051523	020072	000040	
11541					
11542	066552	051440	047510	046125	MTB77: .ASCIZ ' SHOULD HAVE BEEN A HIT IN GROUP '
11543	066560	020104	040510	042526	
11544	066566	041040	042505	020116	
11545	066574	020101	044510	020124	
11546	066602	047111	043440	047522	
11547	066610	050125	000040		
11548					
11549	066614	043101	042524	020122	MTC77: .ASCIZ 'AFTER REFERENCING'<CRLF>'ADDRESS: '
11550	066622	042522	042506	042522	
11551	066630	041516	047111	100107	
11552	066636	042101	051104	051505	
11553	066644	035123	020040	000	
11554					
11555	066651	040	044127	046111	MTD77: .ASCIZ ' WHILE FORCING SELECTION OF GROUP ''
11556	066656	020105	047506	041522	
11557	066664	047111	020107	042523	
11558	066672	042514	052103	047511	
11559	066700	020116	043117	043440	
11560	066706	047522	050125	000040	
11561					
11562	066714	040600	051122	051117	MTA101: .ASCII <CRLF>'ARROR ADRS REG.'<TAB>'ERROR REG.'<TAB>
11563	066722	040440	051104	020123	
11564	066730	042522	027107	042411	
11565	066736	051122	051117	051040	
11566	066744	043505	004456		
11567	066750	054105	042520	052103	.ASCIZ 'EXPECTED ERR.'<TAB>'PATTERN PUT IN MAINT REG.'<CRLF>
11568	066756	042105	042440	051122	
11569	066764	004456	040520	052124	
11570	066772	051105	020116	052520	
11571	067000	020124	047111	046440	
11572	067006	044501	052116	051040	
11573	067014	043505	100056	000	
11574					
11575	067021	200	043101	042524	MTA120: .ASCIZ <CRLF>'AFTER 2ND CYCLE READ '
11576	067026	020122	047062	020104	
11577	067034	054503	046103	020105	
11578	067042	042522	042101	020040	
11579	067050	000			
11580					
11581	067051	200	043101	042524	MTB120: .ASCIZ <CRLF>'AFTER 4TH CYCLE READ '
11582	067056	020122	052064	020110	
11583	067064	054503	046103	020105	
11584	067072	042522	042101	020040	
11585	067100	000			
11586					
11587	067101	200	043101	042524	MTC120: .ASCIZ <CRLF>'AFTER 6TH CYCLE READ '
11588	067106	020122	052066	020110	
11589	067114	054503	046103	020105	
11590	067122	042522	042101	020040	
11591	067130	000			
11592	067131	200	043101	042524	MTD120: .ASCIZ <CRLF>'AFTER 8TH CYCLE READ '
11593	067136	020122	052070	020110	
11594	067144	054503	046103	020105	
11595	067152	042522	042101	020040	

11596	067160	000				
11597						
11598	067161	200	043101	042524	MTE120: .ASCIZ	<CRLF>'AFTER 10TH CYCLE READ '
11599	067166	020122	030061	044124		
11600	067174	041440	041531	042514		
11601	067202	051040	040505	020104		
11602	067210	000				
11603						
11604	067211	200	043101	042524	MTF120: .ASCIZ	<CRLF>'AFTER 12TH CYCLE READ '
11605	067216	020122	031061	044124		
11606	067224	041440	041531	042514		
11607	067232	051040	040505	020104		
11608	067240	000				
11609						
11610	067241	106	047522	020115	MTG120: .ASCIZ	'FROM THE HIT/MISS REG. EXPECTED '
11611	067246	044124	020105	044510		
11612	067254	027524	044515	051523		
11613	067262	051040	043505	020056		
11614	067270	054105	042520	052103		
11615	067276	042105	000040			
11616						
11617	067302	052200	042510	050040	MTA124: .ASCII	<CRLF>'THE PATTERN BEING USED IN THE MAINTENANCE '
11618	067310	052101	042524	047122		
11619	067316	041040	044505	043516		
11620	067324	052440	042523	020104		
11621	067332	047111	052040	042510		
11622	067340	046440	044501	052116		
11623	067346	047105	047101	042503		
11624	067354	040				
11625	067355	122	043505	051511	.ASCIZ	'REGISTER WAS: '
11626	067362	042524	020122	040527		
11627	067370	035123	000040			
11628						
11629	067374	051200	043105	051105	MTA126: .ASCIZ	<CRLF>'REFERENCED ADDRESS:'<TAB>
11630	067402	047105	042503	020104		
11631	067410	042101	051104	051505		
11632	067416	035123	000011			
11633						
11634	067422	040600	051122	051117	MTB126: .ASCIZ	<CRLF>'ERROR ADDRESS REGISTER:'<TAB>
11635	067430	040440	042104	042522		
11636	067436	051523	051040	043505		
11637	067444	051511	042524	035122		
11638	067452	000011				
11639						
11640	067454	050200	052101	042524	MTA131: .ASCIZ	<CRLF>'PATTERN BEING USED IN THE MAINTENANCE REGISTER:'<TAB>
11641	067462	047122	041040	044505		
11642	067470	043516	052440	042523		
11643	067476	020104	047111	052040		
11644	067504	042510	046440	044501		
11645	067512	052116	047105	047101		
11646	067520	042503	051040	043505		
11647	067526	051511	042524	035122		
11648	067534	000011				
11649						
11650	067536	042600	050130	041505	MTB131: .ASCIZ	<CRLF>'EXPECTED ERROR REGISTER:'<TAB>
11651	067544	042524	020104	051105		

11652	067552	047522	020122	042522	
11653	067560	044507	052123	051105	
11654	067566	004472	000		
11655					
11656	067571	200	047507	020124	MTC131: .ASCIZ <CRLF>'GOT ERROR REGISTER:'<TAB>
11657	067576	051105	047522	020122	
11658	067604	042522	044507	052123	
11659	067612	051105	004472	000	
11660					
11661	067617	200	051105	047522	MTA134: .ASCIZ <CRLF>'ERROR ADR REG.'<TAB>'ERROR REG.'<CRLF>
11662	067624	020122	042101	020122	
11663	067632	042522	027107	042411	
11664	067640	051122	051117	051040	
11665	067646	043505	100056	000	
11666					
11667	067653	200	054105	042520	MTA135: .ASCIZ <CRLF>'EXPECTED ERROR REG.: '
11668	067660	052103	042105	042440	
11669	067666	051122	051117	051040	
11670	067674	043505	035056	020040	
11671	067702	000			
11672					
11673	067703	107	052117	042440	MTB135: .ASCIZ 'GOT ERROR REG.: '
11674	067710	051122	051117	051040	
11675	067716	043505	035056	020040	
11676	067724	000			
11677					
11678	067725	200	054105	042520	MTC135: .ASCIZ <CRLF>'EXPECTED ERROR ADR REG.: '
11679	067732	052103	042105	042440	
11680	067740	051122	051117	040440	
11681	067746	051104	051040	043505	
11682	067754	035056	020040	000	
11683					
11684	067761	107	052117	042440	MTD135: .ASCIZ 'GOT ERROR ADR REG.: '
11685	067766	051122	051117	040440	
11686	067774	051104	051040	043505	
11687	070002	035056	020040	000	
11688					
11689					
11690	070007	015	053412	051101	MS01: .ASCII <15><12>/WARNING- THE SIZE OF MEMORY IS DIFFERENT THEN THAT/<CRLF>
11691	070014	044516	043516	020055	
11692	070022	044124	020105	044523	
11693	070030	042532	047440	020106	
11694	070036	042515	047515	054522	
11695	070044	044440	020123	044504	
11696	070052	043106	051105	047105	
11697	070060	020124	044124	047105	
11698	070066	052040	040510	100124	
11699	070074	044440	042116	041511	.ASCIZ / INDICATED BY THE SYSTEM SIZE REGISTER./
11700	070102	052101	042105	041040	
11701	070110	020131	044124	020105	
11702	070116	054523	052123	046505	
11703	070124	051440	055111	020105	
11704	070132	042522	044507	052123	
11705	070140	051105	000056		
11706	070144	005015	044523	042532	MS02: .ASCIZ <15><12>/SIZE REG. ACTUAL/
11707	070152	051040	043505	020056	

11708	070160	020040	020040	041501		
11709	070166	052524	046101	000		
11710	070173	040	020040	020040	MSG3:	.ASCIZ / /
11711	070200	020040	000			
11712	070203	200	050103	020125	MSG1:	.ASCIZ<CRLF> 'CPU UNDER TEST FOUND TO BE A ''
11713	070210	047125	042504	020122		
11714	070216	042524	052123	043040		
11715	070224	052517	042116	052040		
11716	070232	020117	042502	040440		
11717	070240	000040				
11718	070242	041113	030461	042455	MSG2:	.ASCIZ 'KB11-EM'<CRLF>
11719	070250	100115	000			
11720	070253	113	030502	026461	MSG3:	.ASCIZ 'KB11-B/C'<CRLF>
11721	070260	027502	100103	000		
11722	070265	113	030502	026461	MSG4:	.ASCIZ 'KB11-CM ''<CRLF>
11723	070272	046503	020040	020040		
11724	070300	020040	020040	020040		
11725	070306	020040	020040	020040		
11726	070314	000200				
11727	070316	041113	030461	042455	MSG5:	.ASCIZ 'KB11-E'<CRLF>
11728	070324	000200				
11729	070326	005015	047516	046440	EM724:	.ASCIZ <CR><LF>/NO MAP REGISTERS AVAILABLE FOR UNIBUS PARITY ERROR TEST/
11730	070334	050101	051040	043505		
11731	070342	051511	042524	051522		
11732	070350	040440	040526	046111		
11733	070356	041101	042514	043040		
11734	070364	051117	052440	044516		
11735	070372	052502	020123	040520		
11736	070400	044522	054524	042440		
11737	070406	051122	051117	052040		
11738	070414	051505	000124			
11739						
11740						;THESE ARE THE ERROR MESSAGES:
11741						
11742	070420	020101	042522	042506	EM1:	.ASCIZ 'A REFERENCE WHICH SHOULD HAVE BEEN A HIT WAS A MISS.'
11743	070426	042522	041516	020105		
11744	070434	044127	041511	020110		
11745	070442	044123	052517	042114		
11746	070450	044040	053101	020105		
11747	070456	042502	047105	040440		
11748	070464	044040	052111	053440		
11749	070472	051501	040440	046440		
11750	070500	051511	027123	000		
11751						
11752	070505	125	042516	050130	EM2:	.ASCII 'UNEXPECTED ERROR DURING WORST CASE NOISE TEST ON '
11753	070512	041505	042524	020104		
11754	070520	051105	047522	020122		
11755	070526	052504	044522	043516		
11756	070534	053440	051117	052123		
11757	070542	041440	051501	020105		
11758	070550	047516	051511	020105		
11759	070556	042524	052123	047440		
11760	070564	020116				
11761	070566	040503	044103	020105		.ASCII 'CACHE DATA MEMORY.'<CRLF>
11762	070574	040504	040524	046440		
11763	070602	046505	051117	027131		

11764	070610	200				
11765	070611	101	047040	047117		.ASCIZ 'A NON-CACHE DATA PARITY ERROR OCCURRED WHILE TESTING.'
11766	070616	041455	041501	042510		
11767	070624	042040	052101	020101		
11768	070632	040520	044522	054524		
11769	070640	042440	051122	051117		
11770	070646	047440	041503	051125		
11771	070654	042522	020104	044127		
11772	070662	046111	020105	042524		
11773	070670	052123	047111	027107		
11774	070676	000				
11775						
11776	070677	127	051117	052123	EM3:	.ASCII 'WORST CASE NOISE TEST OF THE CACHE DATA MEMORY '
11777	070704	041440	051501	020105		
11778	070712	047516	051511	020105		
11779	070720	042524	052123	047440		
11780	070726	020106	044124	020105		
11781	070734	040503	044103	020105		
11782	070742	040504	040524	046440		
11783	070750	046505	051117	020131		
11784	070756	043200	044501	042514		.ASCIZ <CRLF>/FAILED WHILE GALLOPING 0'S./
11785	070764	020104	044127	046111		
11786	070772	020105	040507	046114		
11787	071000	050117	047111	020107		
11788	071006	023460	027123	000		
11789						
11790	071013	127	051117	052123	EM4:	.ASCII 'WORST CASE NOISE TEST OF THE CACHE DATA MEMORY'
11791	071020	041440	051501	020105		
11792	071026	047516	051511	020105		
11793	071034	042524	052123	047440		
11794	071042	020106	044124	020105		
11795	071050	040503	044103	020105		
11796	071056	040504	040524	046440		
11797	071064	046505	051117	131		
11798	071071	200	040506	046111		.ASCIZ <CRLF>/FAILED WHILE GALLOPING 1'S./
11799	071076	042105	053440	044510		
11800	071104	042514	043440	046101		
11801	071112	047514	044520	043516		
11802	071120	030440	051447	000056		
11803						
11804	071126	042103	054115	052040	EM5:	.ASCIZ 'CDMX TEST FAILURE.'<CRLF>'BAD CACHE GROUP 0 DATA READ.'
11805	071134	051505	020124	040506		
11806	071142	046111	051125	027105		
11807	071150	041200	042101	041440		
11808	071156	041501	042510	043440		
11809	071164	047522	050125	030040		
11810	071172	042040	052101	020101		
11811	071200	042522	042101	000056		
11812						
11813	071206	042103	054115	052040	EM6:	.ASCIZ 'CDMX TEST FAILURE.'<CRLF>'BAD CACHE GROUP 1 DATA READ.'
11814	071214	051505	020124	040506		
11815	071222	046111	051125	027105		
11816	071230	041200	042101	041440		
11817	071236	041501	042510	043440		
11818	071244	047522	050125	030440		
11819	071252	042040	052101	020101		

11820	071260	042522	042101	000056		
11821						
11822	071266	042103	054115	052040	EM7:	.ASCII 'CDMX TEST FAILURE.' <crlf>'bad even="" main="" memory,="" td="" word.'<=""></crlf>'bad>
11823	071274	051505	020124	040506		
11824	071302	046111	051125	027105		
11825	071310	041200	042101	046440		
11826	071316	044501	020116	042515		
11827	071324	047515	054522	020054		
11828	071332	053105	047105	053440		
11829	071340	051117	026104			
11830	071344	042040	052101	020101		.ASCIZ ' DATA READ.'
11831	071352	042522	042101	000056		
11832						
11833	071360	042103	054115	052040	EM10:	.ASCII 'CDMX TEST FAILURE.' <crlf>'bad main="" memory,="" odd="" td="" word.'<=""></crlf>'bad>
11834	071366	051505	020124	040506		
11835	071374	046111	051125	027105		
11836	071402	041200	042101	046440		
11837	071410	044501	020116	042515		
11838	071416	047515	054522	020054		
11839	071424	042117	020104	047527		
11840	071432	042122	054			
11841	071435	040	040504	040524		.ASCIZ ' DATA READ.'
11842	071442	051040	040505	027104		
11843	071450	000				
11844						
11845	071451	120	051101	052111	EM11:	.ASCIZ 'PARITY ERROR IN CACHE DATA MEMORY COUNT PATTERN TEST.'
11846	071456	020131	051105	047522		
11847	071464	020122	047111	041440		
11848	071472	041501	042510	042040		
11849	071500	052101	020101	042515		
11850	071506	047515	054522	041440		
11851	071514	052517	052116	050040		
11852	071522	052101	042524	047122		
11853	071530	052040	051505	027124		
11854	071536	000				
11855						
11856	071537	102	042101	042040	EM12:	.ASCII 'BAD DATA WAS READ IN CACHE MEMORY COUNT PATTERN '
11857	071544	052101	020101	040527		
11858	071552	020123	042522	042101		
11859	071560	044440	020116	040503		
11860	071566	044103	020105	042515		
11861	071574	047515	054522	041440		
11862	071602	052517	052116	050040		
11863	071610	052101	042524	047122		
11864	071616	040				
11865	071617	124	051505	027124		.ASCIZ 'TEST.' <crlf>'but abort="" no="" occurred.'<="" or="" td="" trap=""></crlf>'but>
11866	071624	041200	052125	047040		
11867	071632	020117	051124	050101		
11868	071640	047440	020122	041101		
11869	071646	051117	020124	041517		
11870	071654	052503	051122	042105		
11871	071662	000056				
11872						
11873	071664	040503	044103	020105	EM13:	.ASCII 'CACHE MEMORY COUNT PATTERN TEST.' <crlf>< td=""></crlf><>
11874	071672	042515	047515	054522		
11875	071700	041440	052517	052116		

11876	071706	050040	052101	042524	
11877	071714	047122	052040	051505	
11878	071722	027124	200		
11879	071725	105	051122	051117	.ASCIZ 'ERROR SUMMARY.'
11880	071732	051440	046525	040515	
11881	071740	054522	000056		
11882					
11883	071744	052600	042516	050130	EM14: .ASCIZ <CRLF>'UNEXPECTED PARITY ERROR TRAP.'
11884	071752	041505	042524	020104	
11885	071760	040520	044522	054524	
11886	071766	042440	051122	051117	
11887	071774	052040	040522	027120	
11888	072002	000			
11889					
11890	072003	052	025052	042524	EM15: .ASCIZ '***TEST ABORTED. GOING TO NEXT TEST.***'
11891	072010	052123	040440	047502	
11892	072016	052122	042105	020041	
11893	072024	047507	047111	020107	
11894	072032	047524	047040	054105	
11895	072040	020124	042524	052123	
11896	072046	025056	025052	000	
11897					
11898	072053	103	041501	042510	EM16: .ASCIZ 'CACHE DATA MEMORY DUAL ADDRESS TEST FAILED.'
11899	072060	042040	052101	020101	
11900	072066	042515	047515	054522	
11901	072074	042040	040525	020114	
11902	072102	042101	051104	051505	
11903	072110	020123	042524	052123	
11904	072116	043040	044501	042514	
11905	072124	027104	000		
11906					
11907	072127	103	041501	042510	EM17: .ASCIZ 'CACHE DATA MEMORY BYTE ENABLE LOGIC TEST FAILED.'
11908	072134	042040	052101	020101	
11909	072142	042515	047515	054522	
11910	072150	041040	052131	020105	
11911	072156	047105	041101	042514	
11912	072164	046040	043517	041511	
11913	072172	052040	051505	020124	
11914	072200	040506	046111	042105	
11915	072206	000056			
11916					
11917	072127				EM20-EM17
11918					
11919	072210	040503	044103	020105	EM21: .ASCIZ 'CACHE DATA MEMORY CHIP SELECTION LOGIC TEST FAILED.'
11920	072216	040504	040524	046440	
11921	072224	046505	051117	020131	
11922	072232	044103	050111	051440	
11923	072240	046105	041505	044524	
11924	072246	047117	046040	043517	
11925	072254	041511	052040	051505	
11926	072262	020124	040506	046111	
11927	072270	042105	000056		
11928					
11929	072274	042101	051104	051505	EM22: .ASCII 'ADDRESS MULTIPLEXER TEST WAS UNABLE TO FORCE'
11930	072302	020123	052515	052114	
11931	072310	050111	042514	042530	

11932	072316	020122	042524	052123	
11933	072324	053440	051501	052440	
11934	072332	040516	046102	020105	
11935	072340	047524	043040	051117	
11936	072346	042503			
11937	072350	040440	050040	051101	.ASCII ' A PARITY ERROR, USING THE '<CRLF>
11938	072356	052111	020131	051105	
11939	072364	047522	026122	052440	
11940	072372	044523	043516	052040	
11941	072400	042510	100040		
11942	072404	040515	047111	042524	.ASCII 'MAINTENANCE REGISTER, ON THE'
11943	072412	040516	041516	020105	
11944	072420	042522	044507	052123	
11945	072426	051105	020054	047117	
11946	072434	052040	042510		
11947	072440	046440	044501	020116	.ASCIIZ ' MAIN MEMORY ADDRESS AND CONTROL LINES.'
11948	072446	042515	047515	054522	
11949	072454	040440	042104	042522	
11950	072462	051523	040440	042116	
11951	072470	041440	047117	051124	
11952	072476	046117	046040	047111	
11953	072504	051505	000056		
11954					
11955	072510	042101	051104	051505	EM23: .ASCII 'ADDRESS MULTIPLEXER, AMX, CPU INPUTS TEST FAILED.'
11956	072516	020123	052515	052114	
11957	072524	050111	042514	042530	
11958	072532	026122	040440	054115	
11959	072540	020054	050103	020125	
11960	072546	047111	052520	051524	
11961	072554	052040	051505	020124	
11962	072562	040506	046111	042105	
11963	072570	056			
11964	072571	200	051105	047522	.ASCIIZ <CRLF>'ERROR ADDRESS REGISTER NOT SET CORRECTLY.'
11965	072576	020122	042101	051104	
11966	072604	051505	020123	042522	
11967	072612	044507	052123	051105	
11968	072620	047040	052117	051440	
11969	072626	052105	041440	051117	
11970	072634	042522	052103	054514	
11971	072642	000056			
11972					
11973		072274			EM24=EM22
11974					
11975		072510			EM25-EM23
11976					
11977	072644	042101	051104	051505	EM26: .ASCII 'ADDRESS MEMORY, ADDRESS COMPARATOR TEST FAILURE.'
11978	072652	020123	042515	047515	
11979	072660	054522	020054	042101	
11980	072666	051104	051505	020123	
11981	072674	047503	050115	051101	
11982	072702	052101	051117	052040	
11983	072710	051505	020124	040506	
11984	072716	046111	051125	027105	
11985	072724	040600	020116	042101	.ASCII <CRLF>'AN ADDRESS WHICH SHOULD HAVE BEEN A HIT WAS'
11986	072732	051104	051505	020123	
11987	072740	044127	041511	020110	

11988	072746	044123	052517	042114	
11989	072754	044040	053101	020105	
11990	072762	042502	047105	040440	
11991	072770	044040	052111	053440	
11992	072776	051501			
11993	073000	040440	046440	051511	.ASCIZ ' A MISS.'
11994	073006	027123	000		
11995					
11996	073011	101	042104	042522	EM27: .ASCII 'ADDRESS MEMORY, ADDRESS COMPARATOR TEST FAILURE.'
11997	073016	051523	046440	046505	
11998	073024	051117	026131	040440	
11999	073032	042104	042522	051523	
12000	073040	041440	046517	040520	
12001	073046	040522	047524	020122	
12002	073054	042524	052123	043040	
12003	073062	044501	052514	042522	
12004	073070	056			
12005	073071	200	047101	040440	.ASCII <CRLF>'AN ADDRESS WHICH SHOULD HAVE BEEN A MISS '
12006	073076	042104	042522	051523	
12007	073104	053440	044510	044103	
12008	073112	051440	047510	046125	
12009	073120	020104	040510	042526	
12010	073126	041040	042505	020116	
12011	073134	020101	044515	051523	
12012	073142	040			
12013	073143	127	051501	040440	.ASCIZ 'WAS A HIT.'
12014	073150	044040	052111	000056	
12015					
12016		072274			EM30-EM22
12017					
12018	073156	042101	051104	051505	EM31: .ASCII 'ADDRESS MULTIPLEXER, AMX, UNIBUS INPUTS TEST FAILED.'
12019	073164	020123	052515	052114	
12020	073172	050111	042514	042530	
12021	073200	026122	040440	054115	
12022	073206	020054	047125	041111	
12023	073214	051525	044440	050116	
12024	073222	052125	020123	042524	
12025	073230	052123	043040	044501	
12026	073236	042514	027104		
12027	073242	042600	051122	051117	.ASCIZ <CRLF>'ERROR ADDRESS REGISTER NOT SET CORRECTLY.'
12028	073250	040440	042104	042522	
12029	073256	051523	051040	043505	
12030	073264	051511	042524	020122	
12031	073272	047516	020124	042523	
12032	073300	020124	047503	051122	
12033	073306	041505	046124	027131	
12034	073314	000			
12035					
12036		072274			EM32 EM22
12037					
12038		073156			EM33 EM31
12039					
12040	073315	101	042104	042522	EM34: .ASCII 'ADDRESS MULTIPLEXER, AMX, DUAL ADDRESS TEST,'<CRLF>
12041	073322	051523	046440	046125	
12042	073330	044524	046120	054105	
12043	073336	051105	020054	046501	

12044	073344	026130	042040	040525	
12045	073352	020114	042101	051104	
12046	073360	051505	020123	042524	
12047	073366	052123	100054		
12048	073372	047117	041440	052520	.ASCIZ 'ON CPU INPUTS, FAILED.'
12049	073400	044440	050116	052125	
12050	073406	026123	043040	044501	
12051	073414	042514	027104	000	
12052					
12053	073421	101	042104	042522	EM35: .ASCII 'ADDRESS MULTIPLEXER, AMX, DUAL ADDRESS TEST,'<CRLF>
12054	073426	051523	046440	046125	
12055	073434	044524	046120	054105	
12056	073442	051105	020054	046501	
12057	073450	026130	042040	040525	
12058	073456	020114	042101	051104	
12059	073464	051505	020123	042524	
12060	073472	052123	100054		
12061	073476	047117	052440	044516	.ASCIZ 'ON UNIBUS INPUTS, FAILED.'
12062	073504	052502	020123	047111	
12063	073512	052520	051524	020054	
12064	073520	040506	046111	042105	
12065	073526	000056			
12066					
12067	073530	042101	051104	051505	EM36: .ASCII 'ADDRESS MEMORY COUNT PATTERN TEST FAILURE,'<CRLF>
12068	073536	020123	042515	047515	
12069	073544	054522	041440	052517	
12070	073552	052116	050040	052101	
12071	073560	042524	047122	052040	
12072	073566	051505	020124	040506	
12073	073574	046111	051125	026105	
12074	073602	200			
12075	073603	116	020117	040520	.ASCIZ 'NO PARITY ERROR OCCURS, BUT CAN NOT GET A HIT.'
12076	073610	044522	054524	042440	
12077	073616	051122	051117	047440	
12078	073624	041503	051125	026123	
12079	073632	041040	052125	041440	
12080	073640	047101	047040	052117	
12081	073646	043440	052105	040440	
12082	073654	044040	052111	000056	.
12083					
12084	073662	042101	051104	051505	EM37: .ASCIZ 'ADDRESS MEMORY COUNT PATTERN TEST, ERROR SUMMARY.'
12085	073670	020123	042515	047515	
12086	073676	054522	041440	052517	
12087	073704	052116	050040	052101	
12088	073712	042524	047122	052040	
12089	073720	051505	026124	042440	
12090	073726	051122	051117	051440	
12091	073734	046525	040515	054522	
12092	073742	000056			
12093					
12094	073744	042101	051104	051505	EM40: .ASCII 'ADDRESS MEMORY COUNT PATTERN TEST FAILURE,'<CRLF>
12095	073752	020123	042515	047515	
12096	073760	054522	041440	052517	
12097	073766	052116	050040	052101	
12098	073774	042524	047122	052040	
12099	074002	051505	020124	040506	

12100	074010	046111	051125	026105	
12101	074016	200			
12102	074017	103	041501	042510	.ASCII 'CACHE MEMORY ADDRESS PARITY ERROR OCCURRED'
12103	074024	046440	046505	051117	
12104	074032	020131	042101	051104	
12105	074040	051505	020123	040520	
12106	074046	044522	054524	042440	
12107	074054	051122	051117	047440	
12108	074062	041503	051125	042522	
12109	074070	104			
12110	074071	040	052101	052040	.ASCIIZ ' AT THE TEST ADDRESS.'
12111	074076	042510	052040	051505	
12112	074104	020124	042101	051104	
12113	074112	051505	027123	000	
12114					
12115	074117	101	042104	042522	EM41: .ASCII 'ADDRESS MEMORY DUAL ADDRESS TEST FAILED TO GET '
12116	074124	051523	046440	046505	
12117	074132	051117	020131	052504	
12118	074140	046101	040440	042104	
12119	074146	042522	051523	052040	
12120	074154	051505	020124	040506	
12121	074162	046111	042105	052040	
12122	074170	020117	042507	020124	
12123	074176	020101	044510	020124	.ASCII 'A HIT AT A TEST ADDRESS,'<CRLF>
12124	074204	052101	040440	052040	
12125	074212	051505	020124	042101	
12126	074220	051104	051505	026123	
12127	074226	200			
12128	074227	127	044510	042514	.ASCIIZ 'WHILE WRITING THE ADDRESS MEMORY LOCATIONS.'
12129	074234	053440	044522	044524	
12130	074242	043516	052040	042510	
12131	074250	040440	042104	042522	
12132	074256	051523	046440	046505	
12133	074264	051117	020131	047514	
12134	074272	040503	044524	047117	
12135	074300	027123	000		
12136					
12137	074303	101	042104	042522	EM42: .ASCII 'ADDRESS MEMORY DUAL ADDRESS TEST FAILED TO GET'
12138	074310	051523	046440	046505	
12139	074316	051117	020131	052504	
12140	074324	046101	040440	042104	
12141	074332	042522	051523	052040	
12142	074340	051505	020124	040506	
12143	074346	046111	042105	052040	
12144	074354	020117	042507	124	
12145	074361	101	044040	052111	.ASCII 'A HIT AT A TEST ADDRESS,'<CRLF>
12146	074366	040440	020124	020101	
12147	074374	042524	052123	040440	
12148	074402	042104	042522	051523	
12149	074410	100054			

12150	074412	044127	046111	020105
12151	074420	042522	042101	047111
12152	074426	020107	040502	045503
12153	074434	052040	042510	040440
12154	074442	042104	042522	051523
12155	074450	046440	046505	051117
12156	074456	020131	047514	040503
12157	074464	044524	047117	027123
12158	074472	005200		
12159	074474	052133	044510	020123
12160	074502	051120	041117	042514
12161	074510	020115	044515	044107
12162	074516	020124	042502	041440
12163	074524	051117	042522	052103
12164	074532	042105	041040	020131
12165	074540	041505	020117	034115
12166	074546	034061	026462	056464
12167	074554	000200		

.ASCII 'WHILE READING BACK THE ADDRESS MEMORY LOCATIONS.'

.ASCIIZ '[THIS PROBLEM MIGHT BE CORRECTED BY ECO M8182-4]'

CEKBD-D PDP 11/70-74MP CACHE DIAGNOSTIC PART 2 MACY11 30A(1052) ^{J 4} 16-MAY-79 09:11 PAGE 230
CEKBD.D.P11 16-MAY-79 08:58 MASS BUS TESTER HANDLER

SEQ 0255

12168

12169	074556	042101	051104	051505	EM43:	.ASCII 'ADDRESS MEMORY DUAL ADDRESS TEST FAILURE,'<CRLF>
12170	074564	020123	042515	047515		
12171	074572	054522	042040	040525		
12172	074600	020114	042101	051104		
12173	074606	051505	020123	042524		
12174	074614	052123	043040	044501		
12175	074622	052514	042522	100054		
12176	074630	040503	044103	020105		.ASCIIZ 'CACHE ADDRESS MEMORY PARITY ERROR OCCURRED.'
12177	074636	042101	051104	051505		
12178	074644	020123	042515	047515		
12179	074652	054522	050040	051101		
12180	074660	052111	020131	051105		
12181	074666	047522	020122	041517		
12182	074674	052503	051122	042105		
12183	074702	000056				

12184						
12185	074704	040515	047111	046440	EM44:	.ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED.'
12186	074712	046505	051117	020131		
12187	074720	054502	042524	046440		
12188	074726	051501	020113	042507		
12189	074734	042516	040522	047524		
12190	074742	020122	042524	052123		
12191	074750	043040	044501	042514		
12192	074756	026104				
12193	074760	042040	044517	043516		.ASCII ' DOING CPU DATOB.'<CRLF>
12194	074766	041440	052520	042040		
12195	074774	052101	041117	100056		
12196	075002	020101	040515	047111		.ASCII 'A MAIN MEMORY ADDRESS AND CONTROL LINE '
12197	075010	046440	046505	051117		
12198	075016	020131	042101	051104		
12199	075024	051505	020123	047101		
12200	075032	020104	047503	052116		
12201	075040	047522	020114	044514		
12202	075046	042516	040			
12203	075051	120	051101	052111		.ASCIZ 'PARITY ERROR OCCURRED.'
12204	075056	020131	051105	047522		
12205	075064	020122	041517	052503		
12206	075072	051122	042105	000056		
12207						
12208	075100	040515	047111	046440	EM45:	.ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED.'
12209	075106	046505	051117	020131		
12210	075114	054502	042524	046440		
12211	075122	051501	020113	042507		
12212	075130	042516	040522	047524		
12213	075136	020122	042524	052123		
12214	075144	043040	044501	042514		
12215	075152	026104				
12216	075154	042040	044517	043516		.ASCII ' DOING UNIBUS DATOB.'<CRLF>
12217	075162	052440	044516	052502		
12218	075170	020123	040504	047524		
12219	075176	027102	200			
12220	075201	101	046440	044501		.ASCII 'A MAIN MEMORY ADDRESS AND CONTROL LINE '
12221	075206	020116	042515	047515		
12222	075214	054522	040440	042104		
12223	075222	042522	051523	040440		
12224	075230	042116	041440	047117		
12225	075236	051124	046117	046040		
12226	075244	047111	020105			
12227	075250	040520	044522	054524		.ASCIZ 'PARITY ERROR OCCURRED.'
12228	075256	042440	051122	051117		
12229	075264	047440	041503	051125		
12230	075272	042522	027104	000		
12231						
12232	075277	115	044501	020116	EM46:	.ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED.'
12233	075304	042515	047515	054522		
12234	075312	041040	052131	020105		
12235	075320	040515	045523	043440		
12236	075326	047105	051105	052101		
12237	075334	051117	052040	051505		
12238	075342	020124	040506	046111		
12239	075350	042105	056			

12240	075353	200	051127	047117		.ASCIZ <CRLF>'WRONG BYTE WRITTEN, ON A CPU DATOB.'
12241	075360	020107	054502	042524		
12242	075366	053440	044522	052124		
12243	075374	047105	020054	047117		
12244	075402	040440	041440	052520		
12245	075410	042040	052101	041117		
12246	075416	000056				
12247						
12248	075420	040515	047111	046440	EM47:	.ASCII 'MAIN MEMORY BYTE MASK GENERATOR TEST FAILED.'
12249	075426	046505	051117	020131		
12250	075434	054502	042524	046440		
12251	075442	051501	020113	042507		
12252	075450	042516	040522	047524		
12253	075456	020122	042524	052123		
12254	075464	043040	044501	042514		
12255	075472	027104				
12256	075474	053600	047522	043516		.ASCIZ <CRLF>'WRONG BYTE WRITTEN, ON A UNIBUS DATOB.'
12257	075502	041040	052131	020105		
12258	075510	051127	052111	042524		
12259	075516	026116	047440	020116		
12260	075524	020101	047125	041111		
12261	075532	051525	042040	052101		
12262	075540	041117	000056			
12263						
12264		074704			EM50=EM44	
12265						
12266		075100			EM51=EM45	
12267						
12268		075277			EM52=EM46	
12269						
12270		075420			EM53=EM47	
12271						
12272	075544	040503	044103	020105	EM54:	.ASCII 'CACHE ADDRESS MEMORY POWER UP INVALIDATOR TEST FAILED.'
12273	075552	042101	051104	051505		
12274	075560	020123	042515	047515		
12275	075566	054522	050040	053517		
12276	075574	051105	052440	020120		
12277	075602	047111	040526	044514		
12278	075610	040504	047524	020122		
12279	075616	042524	052123	043040		
12280	075624	044501	042514	027104		
12281	075632	041600	041501	042510		.ASCII <CRLF>'CACHE DATA OR ADDRESS MEMORY PARITY '
12282	075640	042040	052101	020101		
12283	075646	051117	040440	042104		
12284	075654	042522	051523	046440		
12285	075662	046505	051117	020131		
12286	075670	040520	044522	054524		
12287	075676	040				
12288	075677	105	051122	051117		.ASCIZ 'ERROR DETECTED.'
12289	075704	042040	052105	041505		
12290	075712	042524	027104	000		
12291	075717	103	051103	041440	EM55:	.ASCIZ /CCR COULD NOT BE CLEARED/
12292	075724	052517	042114	047040		
12293	075732	052117	041040	020105		
12294	075740	046103	040505	042522		
12295	075746	000104				

12296	075750	053111	051523	024040	EM56: .ASCIZ /IVSS (BIT 14) COULD NOT BE SET IN CCR/
12297	075756	044502	020124	032061	
12298	075764	020051	047503	046125	
12299	075772	020104	047516	020124	
12300	076000	042502	051440	052105	
12301	076006	044440	020116	041503	
12302	076014	000122			
12303	076016	053111	051523	041440	EM57: .ASCIZ /IVSS COULD NOT BE CLEARED IN CCR/
12304	076024	052517	042114	047040	
12305	076032	052117	041040	020105	
12306	076040	046103	040505	042522	
12307	076046	020104	047111	041440	
12308	076054	051103	000		
12309	076057	126	044523	020125	EM60: .ASCIZ /VSIU (BIT '3) COULD NOT BE SET/
12310	076064	041050	052111	030440	
12311	076072	024463	041440	052517	
12312	076100	042114	047040	052117	
12313	076106	041040	020105	042523	
12314	076114	000124			
12315	076116	041526	050111	042040	EM61: .ASCIZ /VCIP DID NOT CLEAR AFTER CACHE FLUSH (ON SETTING VSIU)/
12316	076124	042111	047040	052117	
12317	076132	041440	042514	051101	
12318	076140	040440	052106	051105	
12319	076146	041440	041501	042510	
12320	076154	043040	052514	044123	
12321	076162	024040	047117	051440	
12322	076170	052105	044524	043516	
12323	076176	053040	044523	024525	
12324	076204	000			
12325	076205	126	044523	020125	EM62: .ASCIZ /VSIU COULD NOT BE CLEARED/
12326	076212	047503	046125	020104	
12327	076220	047516	020124	042502	
12328	076226	041440	042514	051101	
12329	076234	042105	000		
12330	076237	126	044503	020120	EM63: .ASCIZ /VCIP DID NOT SET WHEN CACHE FLUSH BIT WAS SET/
12331	076244	044504	020104	047516	
12332	076252	020124	042523	020124	
12333	076260	044127	047105	041440	
12334	076266	041501	042510	043040	
12335	076274	052514	044123	041040	
12336	076302	052111	053440	051501	
12337	076310	051440	052105	000	
12338	076315	126	044523	020125	EM64: .ASCIZ /VSIU DID NOT SWITCH WHEN CACHE FLUSH BIT WAS SET/
12339	076322	044504	020104	047516	
12340	076330	020124	053523	052111	
12341	076336	044103	053440	042510	
12342	076344	020116	040503	044103	
12343	076352	020105	046106	051525	
12344	076360	020110	044502	020124	
12345	076366	040527	020123	042523	
12346	076374	000124			
12347	076376	051526	052511	051440	EM65: .ASCIZ /VSIU SWITCHED WHEN CACHE FLUSH WAS DONE ,WITH IVSS SET/
12348	076404	044527	041524	042510	
12349	076412	020104	044127	047105	
12350	076420	041440	041501	042510	
12351	076426	043040	052514	044123	

12352	076434	053440	051501	042040	
12353	076442	047117	026105	044527	
12354	076450	044124	044440	051526	
12355	076456	020123	042523	000124	
12356	076464	042524	052123	042055	EM66: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12357	076472	052101	020101	042522	
12358	076500	042506	042522	041516	
12359	076506	020105	047516	020124	
12360	076514	020101	044515	051523	
12361	076522	005015	040526	044514	.ASCIZ <15><12>/VALID STORE NOT SWITCHED ON CACHE FLUSH/
12362	076530	020104	052123	051117	
12363	076536	020105	047516	020124	
12364	076544	053523	052111	044103	
12365	076552	042105	047440	020116	
12366	076560	040503	044103	020105	
12367	076566	046106	051525	000110	
12368	076574	042524	052123	042055	EM67: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12369	076602	052101	020101	042522	
12370	076610	042506	042522	041516	
12371	076616	020105	047516	020124	
12372	076624	020101	044515	051523	
12373	076632	005015	040526	044514	.ASCIZ <15><12>/VALID STORE NOT INVALIDATED ON CACHE FLUSH/
12374	076640	020104	052123	051117	
12375	076646	020105	047516	020124	
12376	076654	047111	040526	044514	
12377	076662	040504	042524	020104	
12378	076670	047117	041440	041501	
12379	076676	042510	043040	052514	
12380	076704	044123	000		
12381	076707	124	051505	026524	EM70: .ASCII /TEST-DATA REFERENCE NOT A HIT/
12382	076714	040504	040524	051040	
12383	076722	043105	051105	047105	
12384	076730	042503	047040	052117	
12385	076736	040440	044040	052111	
12386	076744	005015	051106	046517	.ASCIZ <15><12>/FROM THE GROUP AND VALID STORE BEING CHECKED/
12387	076752	052040	042510	043440	
12388	076760	047522	050125	040440	
12389	076766	042116	053040	046101	
12390	076774	042111	051440	047524	
12391	077002	042522	041040	044505	
12392	077010	043516	041440	042510	
12393	077016	045503	042105	000	
12394	077023	104	052101	020101	EM71: .ASCIZ /DATA ERROR ON READING CACHED LOCATION/
12395	077030	051105	047522	020122	
12396	077036	047117	051040	040505	
12397	077044	044504	043516	041440	
12398	077052	041501	042510	020104	
12399	077060	047514	040503	044524	
12400	077066	047117	000		
12401	077071	124	051505	026524	EM72: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12402	077076	040504	040524	051040	
12403	077104	043105	051105	047105	
12404	077112	042503	047040	052117	
12405	077120	040440	046440	051511	
12406	077126	123			
12407	077127	015	041412	041501	.ASCIZ <15><12>/CACHE DOES NOT TURN OFF, WHEN FLUSH DONE WITH IVSS SET/

12408	077134	042510	042040	042517
12409	077142	020123	047516	020124
12410	077150	052524	047122	047440
12411	077156	043106	020054	044127
12412	077164	047105	043040	052514
12413	077172	044123	042040	047117
12414	077200	020105	044527	044124
12415	077206	044440	051526	020123
12416	077214	042523	000124	
12417	077220	042524	052123	042055
12418	077226	052101	020101	042522
12419	077234	042506	042522	041516
12420	077242	020105	047516	020124
12421	077250	020101	044510	124
12422	077255	015	041412	041501
12423	077262	042510	042040	042517
12424	077270	020123	047516	020124
12425	077276	052524	047122	047440
12426	077304	020116	043101	042524
12427	077312	020122	052524	047122
12428	077320	047111	020107	043117
12429	077326	000106		
12430	077330	042524	052123	042055
12431	077336	052101	020101	042522
12432	077344	042506	042522	041516
12433	077352	020105	047516	020124
12434	077360	020101	044515	051523
12435	077366	005015	040503	044103
12436	077374	020105	054502	040520
12437	077402	051523	042040	042111
12438	077410	047040	052117	043040
12439	077416	051117	042503	040440
12440	077424	046440	051511	000123
12441	077432	042524	052123	042055
12442	077440	052101	020101	042522
12443	077446	042506	042522	041516
12444	077454	020105	047516	020124
12445	077462	020101	044515	051523
12446	077470	005015	040503	044103
12447	077476	020105	054502	040520
12448	077504	051523	042040	042111
12449	077512	047040	052117	044440
12450	077520	053116	046101	042111
12451	077526	052101	020105	040503
12452	077534	044103	042105	042040
12453	077542	052101	000101	
12454	077546	042524	052123	042055
12455	077554	052101	020101	042522
12456	077562	042506	042522	041516
12457	077570	020105	047516	020124
12458	077576	020101	044515	051523
12459	077604	005015	051501	041122
12460	077612	042040	042111	047040
12461	077620	052117	043040	051117
12462	077626	042503	040440	046440
12463	077634	051511	020123	047117

EM73: .ASCII /TEST-DATA REFERENCE NOT A HIT/

.ASCIZ <15><12>/CACHE DOES NOT TURN ON AFTER TURNING OFF/

EM74: .ASCII /TEST-DATA REFERENCE NOT A MISS/

.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/

12464 077642 052040 042510 047440
12465 077650 042520 040522 042116
12466 077656 000
12467 077657 124 051505 026524
12468 077664 040504 040524 051040
12469 077672 043105 051105 047105
12470 077700 042503 047040 052117
12471 077706 040440 046440 051511
12472 077714 123
12473 077715 015 041412 041501
12474 077722 042510 020104 050117
12475 077730 051105 047101 020104
12476 077736 047516 020124 047111
12477 077744 040526 044514 040504
12478 077752 042524 020104 047117
12479 077760 040440 051123 020102
12480 077766 054105 041505 052125
12481 077774 047511 100116
12482 100000 052133 044510 020123
12483 100006 051120 041117 042514
12484 100014 020115 044515 044107
12485 100022 020124 042502 041440
12486 100030 051117 042522 052103
12487 100036 042105 041040 020131
12488 100044 041505 020117 034115
12489 100052 034061 026462 056464
12490 100060 000200
12491 100062 042524 052123 042055
12492 100070 052101 020101 047503
12493 100076 046125 020104 047516
12494 100104 020124 042502 046440
12495 100112 042101 020105 044510
12496 100120 000124
12497 100122 047516 050040 051101
12498 100130 052111 020131 051105
12499 100136 047522 020122 051124
12500 100144 050101 047440 020116
12501 100152 040526 044514 020104
12502 100160 052123 051117 020105
12503 100166 040520 044522 054524
12504 100174 042440 051122 051117
12505 100202 000
12506 100203 124 051505 026524
12507 100210 040504 040524 051055
12508 100216 043105 051105 047105
12509 100224 042503 043440 053111
12510 100232 047111 020107 040526
12511 100240 044514 020104 052123
12512 100246 051117 020105 040520
12513 100254 044522 054524
12514 100260 005015 051105 047522
12515 100266 020122 040527 020123
12516 100274 047516 020124 020101
12517 100302 044515 051523 000
12518 100307 106 050126 020105
12519 100314 044504 020104 047516

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>

EM100: .ASCIZ /TEST-DATA COULD NOT BE MADE HIT/

EM103: .ASCIZ /NO PARITY ERROR TRAP ON VALID STORE PARITY ERROR/

EM104: .ASCII /TEST-DATA-REFERENCE GIVING VALID STORE PARITY/

.ASCIZ <15><12>/ERROR WAS NOT A MISS/

EM105: .ASCIZ /FVPE DID NOT GET CLEARED AFTER VSPE OCCURED/

12520	100322	020124	042507	020124	
12521	100330	046103	040505	042522	
12522	100336	020104	043101	042524	
12523	100344	020122	051526	042520	
12524	100352	047440	041503	051125	
12525	100360	042105	000		
12526	100363	126	046101	042111	EM106: .ASCIZ /VALID-STORE-PARITY-ERROR BIT DID NOT SET IN CCR ON VSPE/
12527	100370	051455	047524	042522	
12528	100376	050055	051101	052111	
12529	100404	026531	051105	047522	
12530	100412	020122	044502	020124	
12531	100420	044504	020104	047516	
12532	100426	020124	042523	020124	
12533	100434	047111	041440	051103	
12534	100442	047440	020116	051526	
12535	100450	042520	000		
12536	100453	106	051501	020124	EM107: .ASCII /FAST ADDRESS MEMORY PARITY ERROR BITS (4,5) NOT/
12537	100460	042101	051104	051505	
12538	100466	020123	042515	047515	
12539	100474	054522	050040	051101	
12540	100502	052111	020131	051105	
12541	100510	047522	020122	044502	
12542	100516	051524	024040	026064	
12543	100524	024465	047040	052117	
12544	100532	005015	042523	020124	.ASCIZ <15><12>/SET CORRECTLY IN MSER ON VSPE/
12545	100540	047503	051122	041505	
12546	100546	046124	020131	047111	
12547	100554	046440	042523	020122	
12548	100562	047117	053040	050123	
12549	100570	000105			
12550	100572	051526	052511	051440	EM110: .ASCIZ /VSIU SWITCHED ON VSPE/
12551	100600	044527	041524	042510	
12552	100606	020104	047117	053040	
12553	100614	050123	000105		
12554	100620	042515	047515	054522	EM111: .ASCIZ /MEMORY SYSTEM ERROR REGISTER COULD NOT BE CLEARED/
12555	100626	051440	051531	042524	
12556	100634	020115	051105	047522	
12557	100642	020122	042522	044507	
12558	100650	052123	051105	041440	
12559	100656	052517	042114	047040	
12560	100664	052117	041040	020105	
12561	100672	046103	040505	042522	
12562	100700	000104			
12563	100702	051526	042520	041440	EM112: .ASCIZ /VSPE COULD NOT BE CLEARED IN CCR/
12564	100710	052517	042114	047040	
12565	100716	052117	041040	020105	
12566	100724	046103	040505	042522	
12567	100732	020104	047111	041440	
12568	100740	051103	000		
12569	100743	124	051505	026524	EM113: .ASCIZ /TEST-DATA-REFERENCE NOT A HIT/
12570	100750	040504	040524	051055	
12571	100756	043105	051105	047105	
12572	100764	042503	047040	052117	
12573	100772	040440	044040	052111	
12574	101000	000			
12575	101001	124	051505	026524	EM115: .ASCII /TEST-DATA-REFERNECE NOT A MISS/

12576	101006	040504	040524	051055	
12577	101014	043105	051105	042516	
12578	101022	042503	047040	052117	
12579	101030	040440	046440	051511	
12580	101036	123			
12581	101037	015	041412	041501	.ASCIZ <15><12>/CACHE DID NOT TURN OFF ON BACK-TO-BACK FLUSH/
12582	101044	042510	042040	042111	
12583	101052	047040	052117	052040	
12584	101060	051125	020116	043117	
12585	101066	020106	047117	041040	
12586	101074	041501	026513	047524	
12587	101102	041055	041501	020113	
12588	101110	046106	051525	000110	
12589					
12590	101116	054502	020120	044502	EM123: .ASCIZ ?BYP BIT IN KIPDR COULD NOT BE CLEARED?
12591	101124	020124	047111	045440	
12592	101132	050111	051104	041440	
12593	101140	052517	042114	047040	
12594	101146	052117	041040	020105	
12595	101154	046103	040505	042522	
12596	101162	000104			
12597	101164	054502	020120	044502	EM124: .ASCIZ ?BYP BIT IN KIPDR COULD NOT BE SET?
12598	101172	020124	047111	045440	
12599	101200	050111	051104	041440	
12600	101206	052517	042114	047040	
12601	101214	052117	041040	020105	
12602	101222	042523	000124		
12603	101226	042524	052123	042055	EM125: .ASCIZ /TEST-DATA COULD NOT BE MADE HIT/
12604	101234	052101	020101	047503	
12605	101242	046125	020104	047516	
12606	101250	020124	042502	046440	
12607	101256	042101	020105	044510	
12608	101264	000124			
12609	101266	042524	052123	042055	EM126: .ASCII /TEST-DATA REFERENCE NOT A MISS/
12610	101274	052101	020101	042522	
12611	101302	042506	042522	041516	
12612	101310	020105	047516	020124	
12613	101316	020101	044515	051523	
12614	101324	005015	040503	044103	.ASCIZ <15><12>/CACHED DATA WAS NOT FORCED A MISS ON VIRTUAL PAGE BYPASS/
12615	101332	042105	042040	052101	
12616	101340	020101	040527	020123	
12617	101346	047516	020124	047506	
12618	101354	041522	042105	040440	
12619	101362	046440	051511	020123	
12620	101370	047117	053040	051111	
12621	101376	052524	046101	050040	
12622	101404	043501	020105	054502	
12623	101412	040520	051523	000	
12624	101417	124	051505	020124	EM127: .ASCII /TEST DATA REFERENCE NOT A MISS/
12625	101424	040504	040524	051040	
12626	101432	043105	051105	047105	
12627	101440	042503	047040	052117	
12628	101446	040440	046440	051511	
12629	101454	123			
12630	101455	015	041412	041501	.ASCIZ <15><12>/CACHED DATA WAS NOT INVALIDATED ON VIRTUAL PAGE BYPASS/
12631	101462	042510	020104	040504	

12632	101470	040524	053440	051501	
12633	101476	047040	052117	044440	
12634	101504	053116	046101	042111	
12635	101512	052101	042105	047440	
12636	101520	020116	044526	052122	
12637	101526	040525	020114	040520	
12638	101534	042507	041040	050131	
12639	101542	051501	000123		
12640	101546	054502	020120	044502	EM130: .ASCIZ ?BYP BIT IN SIPDR COULD NOT BE CLEARED?
12641	101554	020124	047111	051440	
12642	101562	050111	051104	041440	
12643	101570	052517	042114	047040	
12644	101576	052117	041040	020105	
12645	101604	046103	040505	042522	
12646	101612	000104			
12647	101614	054502	020120	044502	EM131: .ASCIZ ?BYP BIT IN SIPDR COULD NOT BE SET?
12648	101622	020124	047111	051440	
12649	101630	050111	051104	041440	
12650	101636	052517	042114	047040	
12651	101644	052117	041040	020105	
12652	101652	042523	000124		
12653	101656	054502	020120	044502	EM132: .ASCIZ ?BYP BIT IN UIPDR COULD NOT BE CLEARED?
12654	101664	020124	047111	052440	
12655	101672	050111	051104	041440	
12656	101700	052517	042114	047040	
12657	101706	052117	041040	020105	
12658	101714	046103	040505	042522	
12659	101722	000104			
12660	101724	054502	020120	044502	EM133: .ASCIZ ?BYP BIT IN UIPDR COULD NOT BE SET?
12661	101732	020124	047111	052440	
12662	101740	050111	051104	041440	
12663	101746	052517	042114	047040	
12664	101754	052117	041040	020105	
12665	101762	042523	000124		
12666	101766	040503	044103	020105	EM136: .ASCII 'CACHE ADDRESS MEMORY PARITY LOGIC TEST FAILED.'<CRLF>
12667	101774	042101	051104	051505	
12668	102002	020123	042515	047515	
12669	102010	054522	050040	051101	
12670	102016	052111	020131	047514	
12671	102024	044507	020103	042524	
12672	102032	052123	043040	044501	
12673	102040	042514	027104	200	
12674	102045	125	040516	046102	.ASCII 'UNABLE TO FORCE A PARITY ERROR ON THE LOW BYTE '
12675	102052	020105	047524	043040	
12676	102060	051117	042503	040440	
12677	102066	050040	051101	052111	
12678	102074	020131	051105	047522	
12679	102102	020122	047117	052040	
12680	102110	042510	046040	053517	
12681	102116	041040	052131	020105	
12682	102124	043117	040440	020116	.ASCIZ 'OF AN ADDRESS,'<CRLF>'USING THE MAINTENANCE REGISTER.'
12683	102132	042101	051104	051505	
12684	102140	026123	052600	044523	
12685	102146	043516	052040	042510	
12686	102154	046440	044501	052116	
12687	102162	047105	047101	042503	

12688 102170 051040 043505 051511
12689 102176 042524 027122 000
12690
12691 102203 103 041501 042510
12692 102210 040440 042104 042522
12693 102216 051523 046440 046505
12694 102224 051117 020131 040520
12695 102232 044522 054524 046040
12696 102240 043517 041511 052040
12697 102246 051505 020124 040506
12698 102254 046111 042105 056
12699 102261 200 047125 041101
12700 102266 042514 052040 020117
12701 102274 047506 041522 020105
12702 102302 020101 040520 044522
12703 102310 054524 042440 051122
12704 102316 051117 047440 020116
12705 102324 044124 020105 044510
12706 102332 044107 041040 052131
12707 102340 020105
12708 102342 043117 040440 020116
12709 102350 042101 051104 051505
12710 102356 026123 052600 044523
12711 102364 043516 052040 042510
12712 102372 046440 044501 052116
12713 102400 047105 047101 042503
12714 102406 051040 043505 051511
12715 102414 042524 027122 000
12716
12717 102421
12718 102421 115 044501 020116
12719 102426 042515 047515 054522
12720 102434 042040 052101 020101
12721 102442 040520 044522 054524
12722 102450 041440 042510 045503
12723 102456 051105 020123 042524
12724 102464 052123 043040 044501
12725 102472 042514 027104
12726 102476 052600 040516 046102
12727 102504 020105 047524 043040
12728 102512 051117 042503 040440
12729 102520 050040 051101 052111
12730 102526 020131 051105 047522
12731 102534 026122 052440 044523
12732 102542 043516 040
12733 102545 124 042510 046440
12734 102552 044501 052116 047105
12735 102560 047101 042503 051040
12736 102566 043505 051511 042524
12737 102574 026122 200
12738 102577 101 020124 044124
12739 102604 020105 040515 047111
12740 102612 046440 046505 051117
12741 102620 020131 053105 047105
12742 102626 053440 051117 026104
12743 102634 046040 053517 041040

EM137: .ASCII 'CACHE ADDRESS MEMORY PARITY LOGIC TEST FAILED.'

.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR ON THE HIGH BYTE '

.ASCIIZ 'OF AN ADDRESS,'<CRLF>'USING THE MAINTENANCE REGISTER.'

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 '

12744	102642	052131	026105	050040	
12745	102650	051101	052111	020131	
12746	102656	044103	041505	042513	.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '
12747	102664	026122	020200	042522	
12748	102672	042101	047111	020107	
12749	102700	020101	040504	040524	
12750	102706	050040	052101	042524	
12751	102714	047122	053440	044510	
12752	102722	044103	040		
12753	102725	123	047510	046125	.ASCIZ 'SHOULD HAVE CAUSED AN ERROR.'
12754	102732	020104	040510	042526	
12755	102740	041440	052501	042523	
12756	102746	020104	047101	042440	
12757	102754	051122	051117	000056	
12758					
12759	102762				EM'41:
12760	102762	040515	047111	046440	.ASCII 'MAIN MEMORY DATA PARITY CHECKERS TEST FAILED.'
12761	102770	046505	051117	020131	
12762	102776	040504	040524	050040	
12763	103004	051101	052111	020131	
12764	103012	044103	041505	042513	
12765	103020	051522	052040	051505	
12766	103026	020124	040506	046111	
12767	103034	042105	056		
12768	103037	200	047125	041101	.ASCII <CRLF> 'UNABLE TO FORCE A PARITY ERROR, USING '
12769	103044	042514	052040	020117	
12770	103052	047506	041522	020105	
12771	103060	020101	040520	044522	
12772	103066	054524	042440	051122	
12773	103074	051117	020054	051525	
12774	103102	047111	020107		
12775	103106	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
12776	103114	047111	042524	040516	
12777	103122	041516	020105	042522	
12778	103130	044507	052123	051105	
12779	103136	100054			
12780	103140	052101	052040	042510	.ASCII 'AT THE MAIN MEMORY ODD WORD, LOW BYTE, PARITY '
12781	103146	046440	044501	020116	
12782	103154	042515	047515	054522	
12783	103162	047440	042104	053440	
12784	103170	051117	026104	046040	
12785	103176	053517	041040	052131	
12786	103204	026105	050040	051101	
12787	103212	052111	020131		
12788	103216	044103	041505	042513	.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '
12789	103224	026122	020200	042522	
12790	103232	042101	047111	020107	
12791	103240	020101	040504	040524	
12792	103246	050040	052101	042524	
12793	103254	047122	053440	044510	
12794	103262	044103	040		
12795	103265	123	047510	046125	.ASCIZ 'SHOULD HAVE CAUSED AN ERROR.'
12796	103272	020104	040510	042526	
12797	103300	041440	052501	042523	
12798	103306	020104	047101	042440	
12799	103314	051122	051117	000056	

12800					
12801	103322				EM142:
12802	103322	040515	047111	046440	.ASCII 'MAIN MEMORY DATA PARITY CHECKERS TEST FAILED.'
12803	103330	046505	051117	020131	
12804	103336	040504	040524	050040	
12805	103344	051101	052111	020131	
12806	103352	044103	041505	042513	
12807	103360	051522	052040	051505	
12808	103366	020124	040506	046111	
12809	103374	042105	056		
12810	103377	200	047125	041101	.ASCII <CRLF> 'UNABLE TO FORCE A PARITY ERROR, USING '
12811	103404	042514	052040	020117	
12812	103412	047506	041522	020105	
12813	103420	020101	040520	044522	
12814	103426	054524	042440	051122	
12815	103434	051117	020054	051525	
12816	103442	047111	020107		
12817	103446	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
12818	103454	047111	042524	040516	
12819	103462	041516	020105	042522	
12820	103470	044507	052123	051105	
12821	103476	100054			
12822	103500	052101	052040	042510	.ASCII 'AT THE MAIN MEMORY EVEN WORD, HIGH BYTE, PARITY '
12823	103506	046440	044501	020116	
12824	103514	042515	047515	054522	
12825	103522	042440	042526	020116	
12826	103530	047527	042122	020054	
12827	103536	044510	044107	041040	
12828	103544	052131	026105	050040	
12829	103552	051101	052111	020131	
12830	103560	044103	041505	042513	.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '
12831	103566	026122	020200	042522	
12832	103574	042101	047111	020107	
12833	103602	020101	040504	040524	
12834	103610	050040	052101	042524	
12835	103616	047122	053440	044510	
12836	103624	044103	040		
12837	103627	123	047510	046125	.ASCII 'SHOULD HAVE CAUSED AN ERROR.'
12838	103634	020104	040510	042526	
12839	103642	041440	052501	042523	
12840	103650	020104	047101	042440	
12841	103656	051122	051117	000056	
12842					
12843	103664				EM143:
12844	103664	040515	047111	046440	.ASCII 'MAIN MEMORY DATA PARITY CHECKERS TEST FAILED.'
12845	103672	046505	051117	020131	
12846	103700	040504	040524	050040	
12847	103706	051101	052111	020131	
12848	103714	044103	041505	042513	
12849	103722	051522	052040	051505	
12850	103730	020124	040506	046111	
12851	103736	042105	056		
12852	103741	200	047125	041101	.ASCII <CRLF> 'UNABLE TO FORCE A PARITY ERROR, USING '
12853	103746	042514	052040	020117	
12854	103754	047506	041522	020105	
12855	103762	020101	040520	044522	

12856	103770	054524	042440	051122	
12857	103776	051117	020054	051525	
12858	104004	047111	020107		
12859	104010	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
12860	104016	047111	042524	040516	
12861	104024	041516	020105	042522	
12862	104032	044507	052123	051105	
12863	104040	100054			
12864	104042	052101	052040	042510	.ASCII 'AT THE MAIN MEMORY ODD WORD, HIGH BYTE, PARITY '
12865	104050	046440	044501	020116	
12866	104056	042515	047515	054522	
12867	104064	047440	042104	053440	
12868	104072	051117	026104	044040	
12869	104100	043511	020110	054502	
12870	104106	042524	020054	040520	
12871	104114	044522	054524	040	
12872	104121	103	042510	045503	.ASCII 'CHECKER,'<CRLF>' READING A DATA PATTERN WHICH '
12873	104126	051105	100054	051040	
12874	104134	040505	044504	043516	
12875	104142	040440	042040	052101	
12876	104150	020101	040520	052124	
12877	104156	051105	020116	044127	
12878	104164	041511	020110		
12879	104170	044123	052517	042114	.ASCIIZ 'SHOULD HAVE CAUSED AN ERROR.'
12880	104176	044040	053101	020105	
12881	104204	040503	051525	042105	
12882	104212	040440	020116	051105	
12883	104220	047522	027122	000	
12884					
12885	104225				EM144:
12886	104225	103	041501	042510	.ASCII 'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'
12887	104232	042040	052101	020101	
12888	104240	042515	047515	054522	
12889	104246	050040	051101	052111	
12890	104254	020131	044103	041505	
12891	104262	042513	051522	052040	
12892	104270	051505	020124	040506	
12893	104276	046111	042105	056	
12894	104303	200	047125	041101	.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
12895	104310	042514	052040	020117	
12896	104316	047506	041522	020105	
12897	104324	020101	040520	044522	
12898	104332	054524	042440	051122	
12899	104340	051117	020054	051525	
12900	104346	047111	020107		
12901	104352	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
12902	104360	047111	042524	040516	
12903	104366	041516	020105	042522	
12904	104374	044507	052123	051105	
12905	104402	100054			
12906	104404	052101	052040	042510	.ASCII 'AT THE GROUP ZERO,LOW BYTE, DATA PARITY CHECKER,'
12907	104412	043440	047522	050125	
12908	104420	055040	051105	026117	
12909	104426	047514	020127	054502	
12910	104434	042524	020054	040504	
12911	104442	040524	050040	051101	

12912	104450	052111	020131	044103	
12913	104456	041505	042513	026122	
12914	104464	051200	040505	044504	.ASCII <CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
12915	104472	043516	040440	042040	
12916	104500	052101	020101	040520	
12917	104506	052124	051105	020116	
12918	104514	044127	041511	020110	
12919	104522	044123	052517	042114	
12920	104530	044040	053101	020105	
12921	104536	040503	051525	042105	.ASCIZ 'CAUSED AN ERROR.'
12922	104544	040440	020116	051105	
12923	104552	047522	027122	000	
12924					
12925	104557				EM145:
12926	104557	103	041501	042510	.ASCII 'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'
12927	104564	042040	052101	020101	
12928	104572	042515	047515	054522	
12929	104600	050040	051101	052111	
12930	104606	020131	044103	041505	
12931	104614	042513	051522	052040	
12932	104622	051505	020124	040506	
12933	104630	046111	042105	056	
12934	104635	200	047125	041101	.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
12935	104642	042514	052040	020117	
12936	104650	047506	041522	020105	
12937	104656	020101	040520	044522	
12938	104664	054524	042440	051122	
12939	104672	051117	020054	051525	
12940	104700	047111	020107		
12941	104704	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
12942	104712	047111	042524	040516	
12943	104720	041516	020105	042522	
12944	104726	044507	052123	051105	
12945	104734	100054			
12946	104736	052101	052040	042510	.ASCII 'AT THE GROUP ONE,LOW BYTE, DATA PARITY CHECKER,'
12947	104744	043440	047522	050125	
12948	104752	047440	042516	046054	
12949	104760	053517	041040	052131	
12950	104766	026105	042040	052101	
12951	104774	020101	040520	044522	
12952	105002	054524	041440	042510	
12953	105010	045503	051105	054	
12954	105015	200	042522	042101	.ASCII <CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
12955	105022	047111	020107	020101	
12956	105030	040504	040524	050040	
12957	105036	052101	042524	047122	
12958	105044	053440	044510	044103	
12959	105052	051440	047510	046125	
12960	105060	020104	040510	042526	
12961	105066	040			
12962	105067	103	052501	042523	.ASCIZ 'CAUSED AN ERROR.'
12963	105074	020104	047101	042440	
12964	105102	051122	051117	000056	
12965					
12966	105110				EM146:
12967	105110	040503	044103	020105	.ASCII 'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'

12968	105116	040504	040524	046440	
12969	105124	046505	051117	020131	
12970	105132	040520	044522	054524	
12971	105140	041440	042510	045503	
12972	105146	051105	020123	042524	
12973	105154	052123	043040	044501	
12974	105162	042514	027104		
12975	105166	052600	040516	046102	.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
12976	105174	020105	047524	043040	
12977	105202	051117	042503	040440	
12978	105210	050040	051101	052111	
12979	105216	020131	051105	047522	
12980	105224	026122	052440	044523	
12981	105232	043516	040		
12982	105235	124	042510	046440	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>
12983	105242	044501	052116	047105	
12984	105250	047101	042503	051040	
12985	105256	043505	051511	042524	
12986	105264	026122	200		
12987	105267	101	020124	044124	.ASCII 'AT THE GROUP ZERO,HIGH BYTE, DATA PARITY CHECKER,'
12988	105274	020105	051107	052517	
12989	105302	020120	042532	047522	
12990	105310	044054	043511	020110	
12991	105316	054502	042524	020054	
12992	105324	040504	040524	050040	
12993	105332	051101	052111	020131	
12994	105340	044103	041505	042513	
12995	105346	026122			
12996	105350	051200	040505	044504	.ASCII <CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
12997	105356	043516	040440	042040	
12998	105364	052101	020101	040520	
12999	105372	052124	051105	020116	
13000	105400	044127	041511	020110	
13001	105406	044123	052517	042114	
13002	105414	044040	053101	020105	
13003	105422	040503	051525	042105	.ASCIIZ 'CAUSED AN ERROR.'
13004	105430	040440	020116	051105	
13005	105436	047522	027122	000	
13006					
13007	105443				
13008	105443	103	041501	042510	EM147: .ASCII 'CACHE DATA MEMORY PARITY CHECKERS TEST FAILED.'
13009	105450	042040	052101	020101	
13010	105456	042515	047515	054522	
13011	105464	050040	051101	052111	
13012	105472	020131	044103	041505	
13013	105500	042513	051522	052040	
13014	105506	051505	020124	040506	
13015	105514	046111	042105	056	
13016	105521	200	047125	041101	.ASCII <CRLF>'UNABLE TO FORCE A PARITY ERROR, USING '
13017	105526	042514	052040	020117	
13018	105534	047506	041522	020105	
13019	105542	020101	040520	044522	
13020	105550	054524	042440	051122	
13021	105556	051117	020054	051525	
13022	105564	047111	020107		
13023	105570	044124	020105	040515	.ASCII 'THE MAINTENANCE REGISTER,'<CRLF>

13024	105576	047111	042524	040516	
13025	105604	041516	020105	042522	
13026	105612	044507	052123	051105	
13027	105620	100054			
13028	105622	052101	052040	042510	.ASCII 'AT THE GROUP ONE,HIGH BYTE, DATA PARITY CHECKER.'
13029	105630	043440	047522	050125	
13030	105636	047440	042516	044054	
13031	105644	043511	020110	054502	
13032	105652	042524	020054	040504	
13033	105660	040524	050040	051101	
13034	105666	052111	020131	044103	
13035	105674	041505	042513	026122	
13036	105702	051200	040505	044504	.ASCII <CRLF>'READING A DATA PATTERN WHICH SHOULD HAVE '
13037	105710	043516	040440	042040	
13038	105716	052101	020101	040520	
13039	105724	052124	051105	020116	
13040	105732	044127	041511	020110	
13041	105740	044123	052517	042114	
13042	105746	044040	053101	020105	
13043	105754	040503	051525	042105	.ASCIIZ 'CAUSED AN ERROR.'
13044	105762	040440	020116	051105	
13045	105770	047522	027122	000	
13046					
13047	105775	200	047125	054105	EM150: .ASCIIZ <CRLF>'UNEXPECTED CPU ERROR TRAPPED TO VECTOR ERRVEC (4)!'
13048	106002	042520	052103	042105	
13049	106010	041440	052520	042440	
13050	106016	051122	051117	052040	
13051	106024	040522	050120	042105	
13052	106032	052040	020117	042526	
13053	106040	052103	051117	042440	
13054	106046	051122	042526	020103	
13055	106054	032050	020451	000	
13056					
13057	106061	115	051501	020123	EM151: .ASCIIZ 'MASS BUS WRITE HIT DID NOT INVALIDATE THE CACHE.'
13058	106066	052502	020123	051127	
13059	106074	052111	020105	044510	
13060	106102	020124	044504	020104	
13061	106110	047516	020124	047111	
13062	106116	040526	044514	040504	
13063	106124	042524	052040	042510	
13064	106132	041440	041501	042510	
13065	106140	000056			
13066					
13067		106061			EM152=EM151
13068		106061			EM153=EM151
13069					
13070	106142	042504	044526	042503	EM154: .ASCIIZ 'DEVICE ERROR IN THE RS04.'
13071	106150	042440	051122	051117	
13072	106156	044440	020116	044124	
13073	106164	020105	051522	032060	
13074	106172	000056			
13075					
13076	106174	042504	044526	042503	EM155: .ASCIIZ 'DEVICE ERROR IN THE RP04.'
13077	106202	042440	051122	051117	
13078	106210	044440	020116	044124	
13079	106216	020105	050122	032060	

13080	106224	000056				
13081						
13082	106226	042504	044526	042503	EM156:	.ASCIZ 'DEVICE ERROR IN THE MASS BUS TESTER.'
13083	106234	042440	051122	051117		
13084	106242	044440	020116	044124		
13085	106250	020105	040515	051523		
13086	106256	041040	051525	052040		
13087	106264	051505	042524	027122		
13088	106272	000				
13089						
13090						
13091	106273	104	053105	041511	EM160:	.ASCIZ 'DEVICE ERROR IN THE RK05.'
13092	106300	020105	051105	047522		
13093	106306	020122	047111	052040		
13094	106314	042510	051040	030113		
13095	106322	027065	000			
13096						
13097	106325	104	053105	041511	EM161:	.ASCIZ 'DEVICE ERROR IN THE UNIBUS EXERCISER.'
13098	106332	020105	051105	047522		
13099	106340	020122	047111	052040		
13100	106346	042510	052440	044516		
13101	106354	052502	020123	054105		
13102	106362	051105	044503	042523		
13103	106370	027122	000			
13104	106373	125	041502	020102	EM162:	.ASCII /UBCB PE ABORT DOESN'T GO LOW OR/<CRLF>
13105	106400	042520	040440	047502		
13106	106406	052122	042040	042517		
13107	106414	047123	052047	043440		
13108	106422	020117	047514	020127		
13109	106430	051117	200			
13110	106433	111	020124	047504		.ASCII /IT DOESN'T GET TO TMCC E33 OR E33 BAD/<CRLF>
13111	106440	051505	023516	020124		
13112	106446	042507	020124	047524		
13113	106454	052040	041515	020103		
13114	106462	031505	020063	051117		
13115	106470	042440	031463	041040		
13116	106476	042101	200			
13117	106501	117	020122	041125		.ASCII /OR UBCB PARITY ERR DOESN'T GET TO TMCB E53/<CRLF>
13118	106506	041103	050040	051101		
13119	106514	052111	020131	051105		
13120	106522	020122	047504	051505		
13121	106530	023516	020124	042507		
13122	106536	020124	047524	052040		
13123	106544	041515	020102	032505		
13124	106552	100063				
13125	106554	051501	040440	046040		.ASCIZ /AS A LOW OR E53(5) BAD/
13126	106562	053517	047440	020122		
13127	106570	032505	024063	024465		
13128	106576	041040	042101	000		
13129	106603	125	041502	020102	EM163:	.ASCII /UBCB PARITY ERR DOESN'T GO LOW OR IT DOES/<CRLF>
13130	106610	040520	044522	054524		
13131	106616	042440	051122	042040		
13132	106624	042517	047123	052047		
13133	106632	043440	020117	047514		
13134	106640	020127	051117	044440		
13135	106646	020124	047504	051505		

13136	106654	200				
13137	106655	116	052117	043440		.ASCIZ /NOT GET TO DAPE/
13138	106662	052105	052040	020117		
13139	106670	040504	042520	000		
13140	106675	104	050101	020105	EM164:	.ASCIZ /DAPE E11(4) BAD OR TV06 DOESN'T GET TO THE ALU/
13141	106702	030505	024061	024464		
13142	106710	041040	042101	047440		
13143	106716	020122	053124	033060		
13144	106724	042040	042517	047123		
13145	106732	052047	043440	052105		
13146	106740	052040	020117	044124		
13147	106746	020105	046101	000125		
13148	106754	040504	042520	042440	EM165:	.ASCIZ /DAPE E7(1) BAD/
13149	106762	024067	024461	041040		
13150	106770	042101	000			
13151	106773	124	041515	020101	EM166:	.ASCIZ /TMCA SEG+CON+PAR DOESN'T GO LOW ON CCBJ PARITY TRAP/
13152	107000	042523	025507	047503		
13153	107006	025516	040520	020122		
13154	107014	047504	051505	023516		
13155	107022	020124	047507	046040		
13156	107030	053517	047440	020116		
13157	107036	041503	045102	050040		
13158	107044	051101	052111	020131		
13159	107052	051124	050101	000		
13160	107057	124	041515	020102	EM167:	.ASCII /TMCB PART DOESN'T GO LOW OR DOES/<CRLF>
13161	107064	040520	052122	042040		
13162	107072	042517	047123	052047		
13163	107100	043440	020117	047514		
13164	107106	020127	051117	042040		
13165	107114	042517	100123			
13166	107120	047516	020124	042507		.ASCIZ /NOT GET TO UBCB OR UBCB E18(1) BAD/
13167	107126	020124	047524	052440		
13168	107134	041502	020102	051117		
13169	107142	052440	041502	020102		
13170	107150	030505	024070	024461		
13171	107156	041040	042101	000		
13172	107163	124	041515	020101	EM170:	.ASCIZ /TMCA E67(8) DOESN'T GO LOW ON MGMT/
13173	107170	033105	024067	024470		
13174	107176	042040	042517	047123		
13175	107204	052047	043440	020117		
13176	107212	047514	020127	047117		
13177	107220	046440	046507	000124		
13178	107226	046524	040503	042440	EM171:	.ASCIZ /TMCA E67(12) DOESN'T GO LOW ON MGMT/
13179	107234	033466	030450	024462		
13180	107242	042040	042517	047123		
13181	107250	052047	043440	020117		
13182	107256	047514	020127	047117		
13183	107264	046440	046507	000124		
13184	107272	046524	040503	042440	EM172:	.ASCII /TMCA E68(6) DOESN'T GO LOW ON PAR TRP/<CRLF>
13185	107300	034066	033050	020051		
13186	107306	047504	051505	023516		
13187	107314	020124	047507	046040		
13188	107322	053517	047440	020116		
13189	107330	040520	020122	051124		
13190	107336	100120				
13191	107340	051117	042440	032464		.ASCIZ /OR E45(4) BAD/

```
13192 107346 032050 020051 040502
13193 107354 000104
13194 107356 046524 040503 042440 EM173: .ASCIZ /TMCA E68(8) DOESN'T GO LOW ON PAR TRP/
13195 107364 034066 034050 020051
13196 107372 047504 051505 023516
13197 107400 020124 047507 046040
13198 107406 053517 047440 020116
13199 107414 040520 020122 051124
13200 107422 000120
13201 107424 046524 041503 050040 EM435: .ASCII /TMCC PRIORITY CLEAR DIDN'T GO LOW OR DIDN'T/<CRLF>
13202 107432 044522 051117 052111
13203 107440 020131 046103 040505
13204 107446 020122 044504 047104
13205 107454 052047 043440 020117
13206 107462 047514 020127 051117
13207 107470 042040 042111 023516
13208 107476 100124
13209 107500 042507 020124 044124 .ASCIZ /GET THRU TMCA E43(2) ON ABORT CLEAR/
13210 107506 052522 052040 041515
13211 107514 020101 032105 024063
13212 107522 024462 047440 020116
13213 107530 041101 051117 020124
13214 107536 046103 040505 000122
13215 107544 052502 020123 041120 EM174: .ASCIZ /BUS PB DIDN'T GET TO UBCB PE ABORT/
13216 107552 042040 042111 023516
13217 107560 020124 042507 020124
13218 107566 047524 052440 041502
13219 107574 020102 042520 040440
13220 107602 047502 052122 000
13221 107607 125 041502 020102 EM175: .ASCIZ /UBCB PARITY ERR DIDN'T GO LOW ON BUS PB/
13222 107614 040520 044522 054524
13223 107622 042440 051122 042040
13224 107630 042111 023516 020124
13225 107636 047507 046040 053517
13226 107644 047440 020116 052502
13227 107652 020123 041120 000
13228
13229 ;THESE ARE DATA HEADERS:
13230
13231 107657 040 052040 051505 DH1: .ASCIZ ' TEST.<TAB>' GROUP.<TAB>'PHYSICAL ADDR.<TAB>'CALL AT PC.'
13232 107664 027124 020011 051107
13233 107672 052517 027120 050011
13234 107700 054510 044523 040503
13235 107706 020114 042101 051104
13236 107714 004456 040503 046114
13237 107722 040440 020124 041520
13238 107730 000056
13239
13240 107732 020040 042524 052123 DH2: .ASCII ' TEST.<TAB>' GROUP.<TAB>'ERROR ADDR REG.<TAB>'ERROR REG.<TAB>
13241 107740 004456 043440 047522
13242 107746 050125 004456 051105
13243 107754 047522 020122 042101
13244 107762 051104 051040 043505
13245 107770 004456 051105 047522
13246 107776 020122 042522 027107
13247 110004 011
```

13248	110005	122	043105	040440		.ASCIZ 'REF ADDR.' <tab>'TRAP AT PC.'</tab>
13249	110012	042104	027122	052011		
13250	110020	040522	020120	052101		
13251	110026	050040	027103	000		
13252						
13253		107732			DH3=DH2	
13254						
13255		107732			DH4=DH2	
13256						
13257	110033	040	052040	051505	DH5:	.ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>'READ.'</tab></tab>
13258	110040	027124	041411	046101		
13259	110046	020114	052101	050040		
13260	110054	027103	051011	040505		
13261	110062	027104	000			
13262						
13263		110033			DH6=DH5	
13264						
13265		110033			DH7=DH5	
13266						
13267		110033			DH10=DH5	
13268						
13269	110065	040	052040	051505	DH11:	.ASCIZ ' TEST.' <tab>' GROUP.'<tab>'TRAP AT PC.'<tab>'ERROR ADDR REG.'</tab></tab></tab>
13270	110072	027124	020011	051107		
13271	110100	052517	027120	052011		
13272	110106	040522	020120	052101		
13273	110114	050040	027103	042411		
13274	110122	051122	051117	040440		
13275	110130	042104	020122	042522		
13276	110136	027107	000			
13277						
13278	110141	040	052040	051505	DH12:	.ASCII ' TEST.' <tab>' GROUP.'<tab>'CALL AT PC.'<tab>'TEST ADDR.'<tab></tab></tab></tab></tab>
13279	110146	027124	020011	051107		
13280	110154	052517	027120	041411		
13281	110162	046101	020114	052101		
13282	110170	050040	027103	052011		
13283	110176	051505	020124	042101		
13284	110204	051104	004456			
13285	110210	040504	040524	053440		.ASCIZ 'DATA WR. DATA READ.'
13286	110216	027122	042040	052101		
13287	110224	020101	042522	042101		
13288	110232	000056				
13289						
13290	110234	020040	042524	052123	DH13:	.ASCII ' TEST.' <tab>' GROUP.'<tab>'*DATA.'<tab>'*DATA.'<tab></tab></tab></tab></tab>
13291	110242	004456	043440	047522		
13292	110250	050125	004456	042052		
13293	110256	052101	027101	025411		
13294	110264	040504	040524	004456		
13295	110272	051105	047522	020122		.ASCIZ 'ERROR COUNT.'
13296	110300	047503	047125	027124		
13297	110306	000				
13298						
13299	110307	040	052040	051505	DH14:	.ASCII ' TEST.' <tab>'CALL AT PC.'<tab>'ERROR ADDR REG.'</tab></tab>
13300	110314	027124	041411	046101		
13301	110322	020114	052101	050040		
13302	110330	027103	042411	051122		
13303	110336	051117	040440	042104		

13304	110344	020122	042522	027107		
13305	110352	052011	040522	020120	.ASCII	<TAB>'TRAP AT PC.'<TAB>
13306	110360	052101	050040	027103		
13307	110366	011				
13308	110367	105	051122	051117	.ASCIIZ	'ERROR REG.'
13309	110374	051040	043505	000056		
13310						
13311	110402	020040	042524	052123	DH15:	.ASCIIZ ' TEST.'<TAB>'CALL AT PC.'
13312	110410	004456	040503	046114		
13313	110416	040440	020124	041520		
13314	110424	000056				
13315						
13316	110426	020040	042524	052123	DH16:	.ASCII ' TEST.'<TAB>' GROUP.'<TAB>'WROTE.'<TAB>'READ.'<TAB>
13317	110434	004456	043440	047522		
13318	110442	050125	004456	051127		
13319	110450	052117	027105	051011		
13320	110456	040505	027104	011		
13321	110463	101	042104	020122	.ASCIIZ	'ADDR TESTED.'<TAB>'CALL AT PC.'
13322	110470	042524	052123	042105		
13323	110476	004456	040503	046114		
13324	110504	040440	020124	041520		
13325	110512	000056				
13326						
13327	110514	020040	042524	052123	DH17:	.ASCII ' TEST.'<TAB>' GROUP.'<TAB>'ERROR AT PC.'<TAB>'READ.'<TAB>
13328	110522	004456	043440	047522		
13329	110530	050125	004456	051105		
13330	110536	047522	020122	052101		
13331	110544	050040	027103	051011		
13332	110552	040505	027104	011		
13333	110557	111	027116	040411	.ASCIIZ	'IN.'<TAB>'ADDRESS.'
13334	110564	042104	042522	051523		
13335	110572	000056				
13336						
13337		110514			DH20-DH17	
13338						
13339	110574	020040	042524	052123	DH21:	.ASCIIZ ' TEST.'<TAB>'CALL AT PC.'<TAB>'READ.'<TAB>' GROUP.'<TAB>'ADDRESS.'
13340	110602	004456	040503	046114		
13341	110610	040440	020124	041520		
13342	110616	004456	042522	042101		
13343	110624	004456	043440	047522		
13344	110632	050125	004456	042101		
13345	110640	051104	051505	027123		
13346	110646	000				
13347						
13348	110647	040	052040	051505	DH22:	.ASCIIZ ' TEST.'<TAB>'CALL AT PC.'<TAB>'EXPECTED ERROR AT.'
13349	110654	027124	041411	046101		
13350	110662	020114	052101	050040		
13351	110670	027103	042411	050130		
13352	110676	041505	042524	020104		
13353	110704	051105	047522	020122		
13354	110712	052101	000056			
13355						
13356	110716	020040	042524	052123	DH23:	.ASCII ' TEST.'<TAB>'CALL AT PC.'<TAB>'EXPECTED ADRS.'<TAB>
13357	110724	004456	040503	046114		
13358	110732	040440	020124	041520		
13359	110740	004456	054105	042520		

13360	110746	052103	042105	040440	
13361	110754	051104	027123	011	
13362	110761	107	052117	040440	.ASCIZ 'GOT ADRS.' <tab>'ERROR REG.'</tab>
13363	110766	051104	027123	042411	
13364	110774	051122	051117	051040	
13365	111002	043505	000056		
13366					
13367		110647			DH24=DH22
13368					
13369		110716			DH25=DH23
13370					
13371	111006	020040	042524	052123	DH26: .ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>' GROUP.'<tab>'ADDRESS.'</tab></tab></tab>
13372	111014	004456	040503	046114	
13373	111022	040440	020124	041520	
13374	111030	004456	043440	047522	
13375	111036	050125	004456	042101	
13376	111044	051104	051505	027123	
13377	111052	000			
13378					
13379	111053	040	052040	051505	DH27: .ASCII ' TEST.' <tab>'CALL AT PC.'<tab>' GROUP.'<tab>'ESTABLISHED HIT.'</tab></tab></tab>
13380	111060	027124	041411	046101	
13381	111066	020114	052101	050040	
13382	111074	027103	020011	051107	
13383	111102	052517	027120	042411	
13384	111110	052123	041101	044514	
13385	111116	044123	042105	044040	
13386	111124	052111	056		
13387	111127	040	052502	020124	.ASCIZ ' BUT GOT HIT.'
13388	111134	047507	020124	044510	
13389	111142	027124	000		
13390					
13391		110647			DH30=DH22
13392					
13393		110716			DH31=DH23
13394					
13395		110647			DH32=DH22
13396					
13397		110716			DH33=DH23
13398					
13399	111145	040	052040	051505	DH34: .ASCII ' TEST.' <tab>'PC OF CALL.'<tab>'READ.'<tab>'IN ADDRESS.'<tab></tab></tab></tab></tab>
13400	111152	027124	050011	020103	
13401	111160	043117	041440	046101	
13402	111166	027114	051011	040505	
13403	111174	027104	044411	020116	
13404	111202	042101	051104	051505	
13405	111210	027123	011		
13406	111213	105	050130	041505	.ASCIZ 'EXPECTED.'
13407	111220	042524	027104	000	
13408					
13409		111145			DH35=DH34
13410					
13411	111225	040	052040	051505	DH36: .ASCIZ ' TEST.' <tab>'CALL AT PC.'<tab>' GROUP.'<tab>'ADDRESS.'</tab></tab></tab>
13412	111232	027124	041411	046101	
13413	111240	020114	052101	050040	
13414	111246	027103	020011	051107	
13415	111254	052517	027120	040411	

13416	111262	042104	042522	051523	
13417	111270	000056			
13418					
13419	111272	020040	042524	052123	DH37: .ASCII ' TEST.'<TAB>' GROUP.'<TAB>'ERROR COUNT.'<TAB>
13420	111300	004456	043440	047522	
13421	111306	050125	004456	051105	
13422	111314	047522	020122	047503	
13423	111322	047125	021124	011	
13424	111327	052	041040	042101	.ASCIZ '* BAD ADRS.'<TAB>' + BAD ADRS.'
13425	111334	040440	051104	027123	
13426	111342	025411	041040	042101	
13427	111350	040440	051104	027123	
13428	111356	000			
13429					
13430					
13431	111357	040	052040	051505	DH41: .ASCIZ ' TEST.'<TAB>'CALL AT PC.'<TAB>' GROUP.'<TAB>'ADDRESS.'
13432	111364	027124	041411	046101	
13433	111372	020114	052101	050040	
13434	111400	027103	020011	051107	
13435	111406	052517	027120	040411	
13436	111414	042104	042522	051523	
13437	111422	000056			
13438					
13439		111357			DH42=DH41
13440					
13441	111424	020040	042524	052123	DH43: .ASCII ' TEST.'<TAB>'CALL AT PC.'<TAB>'TRAP AT PC.'<TAB>' GROUP.'
13442	111432	004456	040503	046114	
13443	111440	040440	020124	041520	
13444	111446	004456	051124	050101	
13445	111454	040440	020124	041520	
13446	111462	004456	043440	047522	
13447	111470	050125	056		
13448					
13449		111424			DH40-DH43
13450					
13451	111473	040	052040	051505	DH44: .ASCII ' TEST.'<TAB>'CALL AT PC.'<TAB>'TRAP AT PC.'<TAB>
13452	111500	027124	041411	046101	
13453	111506	020114	052101	050040	
13454	111514	027103	052011	040522	
13455	111522	020120	052101	050040	
13456	111530	027103	011		
13457	111533	105	051122	051117	.ASCIZ 'ERROR ADRS REG.'<TAB>'ERROR REG.'
13458	111540	040440	051104	020123	
13459	111546	042522	027107	042411	
13460	111554	051122	051117	051040	
13461	111562	043505	000056		
13462					
13463		111473			DH45=DH44
13464					
13465	111566	020040	042524	052123	DH46: .ASCIZ ' TEST.'<TAB>'CALL AT PC.'
13466	111574	004456	040503	046114	
13467	111602	040440	020124	041520	
13468	111610	000056			
13469					
13470		111566			DH47 DH46
13471					

13472	111473				DH50=DH44					
13473										
13474	111473				DH51=DH44					
13475										
13476	111566				DH52=DH46					
13477										
13478	111566				DH53=DH46					
13479										
13480	111612	020040	042524	052123	DH54:	.ASCIZ	'	TEST.'	<TAB>'CALL AT PC.'	<TAB>'ERROR COUNT.'
13481	111620	004456	040503	046114						
13482	111626	040440	020124	041520						
13483	111634	004456	051105	047522						
13484	111642	020122	047503	047125						
13485	111650	027124	000							
13486	111653	040	050040	020103	DH55:	.ASCIZ	/	PC	CCR/	
13487	111660	020040	041440	051103						
13488	111666	000								
13489	111667	040	050040	020103	DH66:	.ASCIZ	/	PC	CCR	GROUP TST-DATA-ADRS/
13490	111674	020040	041440	051103						
13491	111702	020040	043440	047522						
13492	111710	050125	020040	052040						
13493	111716	052123	042055	052101						
13494	111724	026501	042101	051522						
13495	111732	000								
13496	111733	040	050040	020103	DH71:	.ASCIZ	/	PC	EXPCTD	RECVD LOC/
13497	111740	020040	054105	041520						
13498	111746	042124	020040	042522						
13499	111754	053103	020104	046040						
13500	111762	041517	000							
13501	111765	040	050040	020103	DH107:	.ASCIZ	/	PC	CCR	MSER GROUP EXPCTD-B4.5/
13502	111772	020040	041440	051103						
13503	112000	020040	020040	051515						
13504	112006	051105	020040	043440						
13505	112014	047522	050125	020040						
13506	112022	042440	050130	052103						
13507	112030	026504	032102	032454						
13508	112036	000								
13509	112037	040	050040	020103	DH111:	.ASCIZ	/	PC	MSER/	
13510	112044	020040	046440	042523						
13511	112052	000122								
13512	112054	020040	041520	020040	DH115:	.ASCIZ	/	PC	HITMIS/	
13513	112062	020040	044510	046524						
13514	112070	051511	000							
13515										
13516	112073	040	050040	020103	DH123:	.ASCIZ	?	PC	KIPDR	(KIPDR)?
13517	112100	020040	045440	050111						
13518	112106	051104	020040	045450						
13519	112114	050111	051104	000051						
13520	112122	020040	041520	020040	DH125:	.ASCIZ	/	PC	CCR	PAR-ADR (PAR) (PDR) TST-DATA-ADRS(VA)/
13521	112130	020040	041503	020122						
13522	112136	050040	051101	040455						
13523	112144	051104	020040	024040						
13524	112152	040520	024522	020040						
13525	112160	050050	051104	020051						
13526	112166	052040	052123	042055						
13527	112174	052101	026501	042101						

13528	112202	051522	053050	024501	
13529	112210	000			
13530	112211	040	050040	020103	DH130: .ASCIZ ? PC SIPDR (SIPDR)?
13531	112216	020040	051440	050111	
13532	112224	051104	020040	051450	
13533	112232	050111	051104	000051	
13534	112240	020040	041520	020040	DH132: .ASCIZ ? PC UIPDR (UIPDR)?
13535	112246	020040	044525	042120	
13536	112254	020122	024040	044525	
13537	112262	042120	024522	000	
13538					
13539					
13540	112267	040	052040	051505	DH136: .ASCIZ ' TEST.'<TAB>'CALL AT PC.'<TAB>' GROUP.'<TAB>'ADDRESS.'
13541	112274	027124	041411	046101	
13542	112302	020114	052101	050040	
13543	112310	027103	020011	051107	
13544	112316	052517	027120	040411	
13545	112324	042104	042522	051523	
13546	112332	000056			
13547					
13548		112267			DH137=DH136
13549					
13550	112334	020040	042524	052123	DH140: .ASCIZ ' TEST.'<TAB>'CALL AT PC.'<TAB>'DATA.'<TAB>'ADDRESS.'
13551	112342	004456	040503	046114	
13552	112350	040440	020124	041520	
13553	112356	004456	040504	040524	
13554	112364	004456	042101	051104	
13555	112372	051505	027123	000	
13556					
13557		112334			DH141=DH140
13558					
13559		112334			DH142=DH140
13560					
13561		112334			DH143=DH140
13562					
13563		112334			DH144=DH140
13564					
13565		112334			DH145=DH140
13566					
13567		112334			DH146=DH140
13568					
13569		112334			DH147=DH140
13570					
13571	112377	040	052040	051505	DH150: .ASCIZ ' TEST.'<TAB>'TRAP AT PC.'<TAB>'CALL AT PC.'<TAB>'CPU ERROR REGISTER.'
13572	112404	027124	052011	040522	
13573	112412	020120	052101	050040	
13574	112420	027103	041411	046101	
13575	112426	020114	052101	050040	
13576	112434	027103	041411	052520	
13577	112442	042440	051122	051117	
13578	112450	051040	043505	051511	
13579	112456	042524	027122	000	
13580					
13581	112463	125	044523	043516	DH151: .ASCII 'USING THE RS04.'
13582	112470	052040	042510	051040	
13583	112476	030123	027064		

13584	112502	020040	042524	052123		.ASCIZ	'TEST.<TAB>GROUP.<TAB>ADDRESS.'
13585	112510	004456	051107	052517			
13586	112516	027120	040411	042104			
13587	112524	042522	051523	000056			
13588							
13589	112532	051525	047111	020107	DH152:	.ASCII	'USING THE RP04.'
13590	112540	044124	020105	050122			
13591	112546	032060	056				
13592	112551	040	052040	051505		.ASCIZ	'TEST.<TAB>GROUP.<TAB>ADDRESS.'
13593	112556	027124	043411	047522			
13594	112564	050125	004456	042101			
13595	112572	051104	051505	027123			
13596	112600	000					
13597							
13598	112601	125	044523	043516	DH153:	.ASCII	'USING THE MASS BUS TESTER.'
13599	112606	052040	042510	046440			
13600	112614	051501	020123	052502			
13601	112622	020123	042524	052123			
13602	112630	051105	056				
13603	112633	040	052040	051505		.ASCIZ	'TEST.<TAB>GROUP.<TAB>ADDRESS.'
13604	112640	027124	043411	047522			
13605	112646	050125	004456	042101			
13606	112654	051104	051505	027123			
13607	112662	000					
13608							
13609	112663	040	052040	051505	DH154:	.ASCIZ	'TEST.<TAB>RS4CS2.<TAB>RS4DS.<TAB>RS4ER.'
13610	112670	027124	051011	032123			
13611	112676	051503	027062	051011			
13612	112704	032123	051504	004456			
13613	112712	051522	042464	027122			
13614	112720	000					
13615							
13616	112721	040	052040	051505	DH155:	.ASCIZ	'TEST.<TAB>RP4CS2.<TAB>RP4DS.<TAB>RP4ER.'
13617	112726	027124	051011	032120			
13618	112734	051503	027062	051011			
13619	112742	032120	051504	004456			
13620	112750	050122	042464	027122			
13621	112756	000					
13622							
13623	112757	040	052040	051505	DH156:	.ASCIZ	'TEST.<TAB>RH4CS2.<TAB>RH4ST.<TAB>RH4ER.'
13624	112764	027124	051011	032110			
13625	112772	051503	027062	051011			
13626	113000	032110	052123	004456			
13627	113006	044122	042464	027122			
13628	113014	000					
13629							
13630							
13631	113015	040	052040	051505	DH160:	.ASCIZ	'TEST.<TAB>RK5ER.<TAB>RK5DS.'
13632	113022	027124	051011	032513			
13633	113030	051105	004456	045522			
13634	113036	042065	027123	000			
13635							
13636	113043	040	052040	051505	DH161:	.ASCIZ	'TEST.<TAB>UBECR1.<TAB>UBECR2.'
13637	113050	027124	052411	042502			
13638	113056	051103	027061	052411			
13639	113064	042502	051103	027062			

```

13640 113072 000
13641 113073 105 051122 051117 DH162: .ASCIZ /ERRORPC TEST NUMBER/
13642 113100 041520 052040 051505
13643 113106 020124 052516 041115
13644 113114 051105 000
13645
13646 ;THESE ARE DATA FORMAT DESIGNATORS FOR THE DATA TABLE:
13647
13648 113117 004 004 003 DF1: .BYTE 4,4,3,3
13649 113122 003
13650
13651 113123 004 004 007 DF2: .BYTE 4,4,7,0,3,3
13652 113126 000 003 003
13653
13654 113123 DF3=DF2
13655
13656 113123 DF4=DF2
13657
13658 113131 004 003 000 DF5: .BYTE 4,3,0,5,0,0,0,0
13659 113134 005 000 000
13660 113137 000 000
13661
13662 113131 DF6=DF5
13663
13664 113131 DF7=DF5
13665
13666 113131 DF10=DF5
13667
13668 113141 004 004 003 DF11: .BYTE 4,4,3,7,5,0,5,3,0
13669 113144 007 005 000
13670 113147 005 003 000
13671
13672 113152 004 004 003 DF12: .BYTE 4,4,3,3,0,0
13673 113155 003 000 000
13674
13675 113160 004 004 000 DF13: .BYTE 4,4,0,0,4
13676 113163 000 004
13677
13678 113165 004 003 007 DF14: .BYTE 4,3,7,3,0
13679 113170 003 000
13680
13681 113172 004 003 DF15: .BYTE 4,3
13682
13683 113174 004 004 000 DF16: .BYTE 4,4,0,0,3,3
13684 113177 000 003 003
13685
13686 113202 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
13687 113205 000 005 003
13688 113210 005 005 005
13689 113213 003 005 003
13690 113216 005 003 005
13691 113221 003 005 000
13692 113224 005 000 005
13693 113227 000 005 000
13694
13695 113202 DF20 DF17
    
```


13752	113334	004	003	002	DF44:	.BYTE	4,3,2,7,0,5,2,5,0,5,2,5,0,5,2
13753	113337	007	000	005			
13754	113342	002	005	000			
13755	113345	005	002	005			
13756	113350	000	005	002			
13757							
13758	113334				DF45=DF44		
13759							
13760	113353	004	003	005	DF46:	.BYTE	4,3,5,2,5,0,5,2,5,0,5,2
13761	113356	002	005	000			
13762	113361	005	002	005			
13763	113364	000	005	002			
13764							
13765	113353				DF47=DF46		
13766							
13767	113334				DF50=DF44		
13768							
13769	113334				DF51=DF44		
13770							
13771	113353				DF52=DF46		
13772							
13773	113353				DF53=DF46		
13774							
13775	113367	004	003	004	DF54:	.BYTE	4,3,4
13776	113372	000	000		DF55:	.BYTE	0,0
13777	113372				DF56=DF55		
13778	113372				DF57=DF55		
13779	113372				DF60=DF55		
13780	113372				DF61=DF55		
13781	113372				DF62=DF55		
13782	113372				DF63=DF55		
13783	113372				DF64=DF55		
13784	113372				DF65=DF55		
13785	113374	000	000	000	DF66:	.BYTE	0,0,0,0
13786	113377	000					
13787	113374				DF67=DF66		
13788	113374				DF70=DF66		
13789	113374				DF71=DF66		
13790	113374				DF72=DF66		
13791	113374				DF73=DF66		
13792	113374				DF74=DF66		
13793	113374				DF75=DF66		
13794	113374				DF76=DF66		
13795	113374				DF77=DF66		
13796							
13797	113400	000	000	000	DF100:	.BYTE	0,0,0,0,0,0
13798	113403	000	000	000			
13799	113372				DF103=DF55		
13800	113372				DF104=DF55		
13801	113372				DF105=DF55		
13802	113372				DF106=DF55		
13803	113406	000	000	000	DF107:	.BYTE	0,0,0,0,0
13804	113411	000	000				
13805							
13806	113372				DF110=DF55		
13807	113372				DF111=DF55		

```

13808      113372      DF112=DF55
13809      113374      DF113=DF66
13810      113372      DF115=DF55
13811  113413      000      000      000      DF123: .BYTE 0,0,0
13812  113416      004      003      004      DF136: .BYTE 4,3,4,2
13813  113421      002
13814      113416      DF137=DF136
13815
13816  113422      004      003      000      DF140: .BYTE 4,3,0,2
13817  113425      002
13818
13819      113422      DF141=DF140
13820
13821      113422      DF142=DF140
13822
13823      113422      DF143=DF140
13824
13825      113422      DF144=DF140
13826
13827      113422      DF145=DF140
13828
13829      113422      DF146=DF140
13830
13831      113422      DF147=DF140
13832
13833  113426      004      003      003      DF150: .BYTE 4,3,3,0
13834  113431      000
13835
13836  113432      004      004      007      DF151: .BYTE 4,4,7
13837
13838      113432      DF152=DF151
13839      113432      DF153=DF151
13840
13841  113435      004      000      000      DF154: .BYTE 4,0,0,0
13842  113440      000
13843
13844      113435      DF155=DF154
13845      113435      DF156=DF154
13846      113435      DF157=DF154
13847      113435      DF160=DF154
13848      113435      DF161=DF154
13849
13850
13851      113442      .EVEN
13852
13853      ;THESE ARE DATA TABLES:
13854
13855  113442  001626  001630  001632  DT1: .WORD $TMP0,$TMP1,$TMP2,$ERRPC,0
13856  113450  001520  000000
13857
13858  113454  001626  001642  001632  DT2: .WORD $TMP0,$TMP6,$TMP2,$TMP1,$TMP5,$TMP4,0
13859  113462  001630  001640  001636
13860  113470  000000
13861
13862      113454      DT3-DT2
13863
    
```

13864		113454			DT4-DT2				
13865									
13866	113472	001626	001520	001632	DT5:	.WORD	\$TMP0,\$ERRPC,\$TMP2,MTA5,JJPAT1,JJPAT2,JJPAT3,JJPAT4,0		
13867	113500	065100	026132	026134					
13868	113506	026136	026140	000000					
13869									
13870		113472			DT6-DT5				
13871									
13872		113472			DT7=DT5				
13873									
13874		113472			DT10=DT5				
13875									
13876	113514	001626	001630	001632	DT11:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP4,MTA11,\$TMP3,\$TAB,\$TMP7,\$TMP6,0		
13877	113522	001636	065162	001634					
13878	113530	065076	001644	001642					
13879	113536	000000							
13880									
13881	113540	001626	001630	001520	DT12:	.WORD	\$TMP0,\$TMP1,\$ERRPC,\$TMP3,\$TMP4,\$TMP5,0		
13882	113546	001634	001636	001640					
13883	113554	000000							
13884									
13885	113556	001626	001630	001632	DT13:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,\$TMP4,0		
13886	113564	001634	001636	000000					
13887									
13888	113572	001626	001520	001630	DT14:	.WORD	\$TMP0,\$ERRPC,\$TMP1,\$TMP3,\$TMP4,0		
13889	113600	001634	001636	000000					
13890									
13891	113606	001626	001630	000000	DT15:	.WORD	\$TMP0,\$TMP1,0		
13892									
13893	113614	001626	001630	001632	DT16:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,\$TMP4,\$ERRPC,0		
13894	113622	001634	001636	001520					
13895	113630	000000							
13896									
13897	113632	001626	001630	001632	DT17:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,MTA17,\$TMP4,\$CRLF,MTB17		
13898	113640	001634	065254	001636					
13899	113646	001707	065234						
13900	113652	065227	001640	065227		.WORD	MTA17,\$TMP5,MTA17,\$TMP6,MTA17,\$TMP7,MTA17,\$TMP10		
13901	113660	001642	065227	001644					
13902	113666	065227	001646						
13903	113672	001707	035376	065076		.WORD	\$CRLF,\$MMPAT1,\$TAB,\$MMPAT2,\$TAB,\$MMPAT3,\$TAB,\$MMPAT4,0		
13904	113700	035400	065076	035402					
13905	113706	065076	035404	000000					
13906									
13907	113714	001626	001630	001632	DT20:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,MTA20,\$TMP4,\$CRLF,MTB17		
13908	113722	001634	065263	001636					
13909	113730	001707	065234						
13910	113734	001640	065227	001642		.WORD	\$TMP5,MTA17,\$TMP6,MTA17,\$TMP7,MTA17,\$TMP10,MTA17		
13911	113742	065227	001644	065227					
13912	113750	001646	065227						
13913	113754	001707	035376	065076		.WORD	\$CRLF,\$MMPAT1,\$TAB,\$MMPAT3,\$TAB,\$MMPAT3,\$TAB,\$MMPAT4,0		
13914	113762	035402	065076	035402					
13915	113770	065076	035404	000000					
13916									
13917	113776	001626	001630	001632	DT21:	.WORD	\$TMP0,\$TMP1,\$TMP2,\$TMP3,\$TMP4,MTA21		
13918	114004	001634	001636	065272					
13919	114012	065227	001640	065227		.WORD	MTB21,\$TMP5,MTB21,\$TMP6,MTB21,\$TMP7,MTB21,\$TMP10,\$CRLF		

Address	Cache	Cache	Cache	Cache	DT	Format	Fields
13920	114020	001642	065227	001644			
13921	114026	065227	001646	001707			
13922	114034	033532	065076	033534		.WORD	KKPAT1,\$TAB,KKPAT2,\$TAB,KKPAT3,\$TAB,KKPAT4,0
13923	114042	065076	033536	065076			
13924	114050	033540	000000				
13925							
13926	114054	001626	001520	007466	DT22:	.WORD	\$TMP0,\$ERRPC,XADR2,0
13927	114062	000000					
13928							
13929	114064	001626	001520	007466	DT23:	.WORD	\$TMP0,\$ERRPC,XADR2,\$TMP3,\$TMP1,0
13930	114072	001634	001630	000000			
13931							
13932	114100	001626	001520	010366	DT24:	.WORD	\$TMP0,\$ERRPC,XXADR2,0
13933	114106	000000					
13934							
13935	114110	001626	001520	010366	DT25:	.WORD	\$TMP0,\$ERRPC,XXADR2,\$TMP3,\$TMP1,0
13936	114116	001634	001630	000000			
13937							
13938	114124	001626	001520	001630	DT26:	.WORD	\$TMP0,\$ERRPC,\$TMP1,\$TMP2,0
13939	114132	001632	000000				
13940							
13941	114136	001626	001520	001630	DT27:	.WORD	\$TMP0,\$ERRPC,\$TMP1,\$TMP2,\$TMP4,0
13942	114144	001632	001636	000000			
13943							
13944	114152	001626	001520	011304	DT30:	.WORD	\$TMP0,\$ERRPC,RRADR2,0
13945	114160	000000					
13946							
13947	114162	001626	001520	011304	DT31:	.WORD	\$TMP0,\$ERRPC,RRADR2,\$TMP3,\$TMP1,0
13948	114170	001634	001630	000000			
13949							
13950	114176	001626	001520	012166	DT32:	.WORD	\$TMP0,\$ERRPC,SSADR2,0
13951	114204	000000					
13952							
13953	114206	001626	001520	012166	DT33:	.WORD	\$TMP0,\$ERRPC,SSADR2,\$TMP3,\$TMP1,0
13954	114214	001634	001630	000000			
13955							
13956	114222	001626	001520	001632	DT34:	.WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,\$TMP5,0
13957	114230	001634	001640	000000			
13958							
13959		114222			DT35-D134		
13960							
13961	114236	001626	001520	001632	DT36:	.WORD	\$TMP0,\$ERRPC,\$TMP2,BBADR1,0
13962	114244	014772	000000				
13963							
13964	114250	001626	001630	015012	DT37:	.WORD	\$TMP0,\$TMP1,BBCNT1,BBADR2,BBADR3,0
13965	114256	014776	015002	000000			
13966							
13967							
13968	114264	001626	001520	001632	DT41:	.WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,0
13969	114272	001634	000000				
13970							
13971		114264			DT42=DT41		
13972							
13973	114276	001626	001520	001632	DT43:	.WORD	\$TMP0,\$ERRPC,\$TMP2,\$TMP3,MTA43,\$TMP5,\$TMP7,\$TMP4,0
13974	114304	001634	065357	001640			
13975	114312	001644	001636	000000			

```

13976
13977          114276          DT40=DT43
13978
13979 114320 001626 001520 001662 DT44: .WORD $TMP0,$ERRPC,$TMP16,$TMP3,$TMP5,MTA45,$TMP12,MTB45
13980 114326 001634 001640 065432
13981 114334 001652 065460
13982 114340 001646 065475 001642          .WORD $TMP10,MTC45,$TMP6,MTB45,$TMP11,MTC45,$TMP14,0
13983 114346 065460 001650 065475
13984 114354 001656 000000
13985
13986          114320          DT45=DT44
13987
13988 114360 001626 001652 065432 DT46: .WORD $TMP0,$TMP12,MTA45,$TMP10,MTB45,$TMP6,MTC45
13989 114366 001646 065460 001642
13990 114374 065475
13991 114376 001632 065460 001644          .WORD $TMP2,MTB45,$TMP7,MTC45,$TMP4,0
13992 114404 065475 001636 000000
13993
13994          114360          DT47=DT46
13995
13996 114412 001626 001520 001662 DT50: .WORD $TMP0,$ERRPC,$TMP16,$TMP3,$TMP5,MTA50,$TMP12,MTB45
13997 114420 001634 001640 065510
13998 114426 001652 065460
13999 114432 001646 065475 001642          .WORD $TMP10,MTC45,$TMP6,MTB45,$TMP11,MTC45,$TMP14,0
14000 114440 065460 001650 065475
14001 114446 001656 000000
14002
14003          114412          DT51=DT50
14004
14005 114452 001626 001652 065510 DT52: .WORD $TMP0,$TMP12,MTA50,$TMP10,MTB45,$TMP6,MTC45
14006 114460 001646 065460 001642
14007 114466 065475
14008 114470 001632 065460 001644          .WORD $TMP2,MTB45,$TMP7,MTC45,$TMP4,0
14009 114476 065475 001636 000000
14010
14011          114452          DT53=DT52
14012
14013 114504 001626 001520 001632 DT54: .WORD $TMP0,$ERRPC,$TMP2,0
14014 114512 000000
14015 114514 001520 001556 000000 DT55: .WORD $ERRPC,$REG0,0
14016 114522 001520 001556 001560 DT66: .WORD $ERRPC,$REG0,$REG1,$REG2,0
14017 114530 001562 000000
14018 114534 001520 001556 001560 DT100: .WORD $ERRPC,$REG0,$REG1,$REG2,$REG3,$REG4,0
14019 114542 001562 001564 001566
14020 114550 000000
14021 114552 001520 001556 001560 DT107: .WORD $ERRPC,$REG0,$REG1,$REG2,$REG3,0
14022 114560 001562 001564 000000
14023
14024 114566 001520 001556 001560 DT123: .WORD $ERRPC,$REG0,$REG1,0
14025 114574 000000
14026 114576 001520 001556 001560 DT125: .WORD $ERRPC,$REG0,$REG1,$REG2,$REG3,$REG4,0
14027 114604 001562 001564 001566
14028 114612 000000
14029
14030 114614 001626 001520 001632 DT136: .WORD $TMP0,$ERRPC,$TMP2,$TMP3,0
14031 114622 001634 000000
    
```

```
14032
14033      114614      DT137=DT136
14034
14035  114626  001626  001520  001632  DT140:  .WORD  $TMP0,$ERRPC,$TMP2,$TMP3,0
14036  114634  001634  000000
14037
14038      114626      DT141=DT140
14039
14040      114626      DT142=DT140
14041
14042      114626      DT143=DT140
14043
14044      114626      DT144=DT140
14045
14046      114626      DT145=DT140
14047
14048      114626      DT146=DT140
14049
14050      114626      DT147=DT140
14051
14052  114640  001626  001630  001632  DT150:  .WORD  $TMP0,$TMP1,$TMP2,$TMP3,0
14053  114646  001634  000000
14054
14055  114652  001626  001630  001632  DT151:  .WORD  $TMP0,$TMP1,$TMP2,0
14056  114660  000000
14057
14058      114652      DT152=DT151
14059      114652      DT153=DT151
14060  114662  001626  001630  001632  DT154:  .WORD  $TMP0,$TMP1,$TMP2,$TMP3,0
14061  114670  001634  000000
14062      114662      DT155=DT154
14063      114662      DT156=DT154
14064      114652      DT157=DT151
14065      114652      DT160=DT151
14066      114652      DT161=DT151
14067  114674  001520  001502  000000  DT162:  .WORD  $ERRPC,$STNM,0
14068
14069      114702      TLOC=.
14070      114700      TLOC=-4&TLOC
14071      114704      TLOC=TLOC+4
14072      114704      .=TLOC
14073  114704  001000      *STDAT:  .BLKW  512.
14074
14075
14076
14077  116704  000000  000000  000000  BOTTOM:  .WORD  0,0,0
14078      124704      BOTPRG=BOTTOM+6000
14079      000001      .END
```


DDDONE	025042	4808	4813	4827#						
DDPD	024710	4787	4794#							
DDPER	025006	4784	4815#							
DDPER1	025032	4816	4821#							
DDPU1	024746	4803#								
DDPU2	024772	4805	4810#							
DDPV	024722	4795	4797#							
DDTMP	025036	4786*	4798	4803	4823#					
DD1	024654	4781	4784#							
DF1	113117	617	13648#							
DF10 =	113131	638	13666#							
DF100	113400	1022	1029	1036	13797#					
DF103 =	113372	912	13799#							
DF104 =	113372	919	13800#							
DF105 =	113372	926	13801#							
DF106 =	113372	933	13802#							
DF107	113406	940	13803#							
DF11	113141	641	13668#							
DF110 =	113372	947	13806#							
DF111 =	113372	954	13807#							
DF112 =	113372	961	13808#							
DF113 =	113374	968	13809#							
DF115 =	113372	980	13810#							
DF12	113152	644	13672#							
DF123	113413	1011	1016	1041	1047	1053	1059	13811#		
DF13	113160	647	13675#							
DF136	113416	1067	13812#	13814						
DF137 =	113416	1070	13814#							
DF14	113165	650	13678#							
DF140	113422	1073	13816#	13819	13821	13823	13825	13827	13829	13831
DF141 =	113422	1076	13819#							
DF142 =	113422	1079	13821#							
DF143 =	113422	1082	13823#							
DF144 =	113422	1085	13825#							
DF145 =	113422	1088	13827#							
DF146 =	113422	1091	13829#							
DF147 =	113422	1094	13831#							
DF15	113172	653	13681#							
DF150	113426	1097	13833#							
DF151	113432	1100	13836#	13838	13839					
DF152 =	113432	1103	13838#							
DF153 =	113432	1106	13839#							
DF154	113435	1109	13841#	13844	13845	13846	13847	13848		
DF155 =	113435	1112	13844#							
DF156 =	113435	1115	13845#							
DF157 =	113435	13846#								
DF16	113174	656	13683#							
DF160 =	113435	1121	13847#							
DF161 =	113435	1124	13848#							
DF17	113202	659	13686#	13695						
DF2	113123	620	13651#	13654	13656					
DF20 =	113202	662	13695#							
DF21	113232	665	13697#							
DF22	113260	668	13706#	13711	13721	13725				
DF23	113263	671	13708#	13713	13723	13727				
DF24 =	113260	674	13711#							

EE10	023344	4508	4512#	4567	
EE11	023374	4519#	4522		
EE12	023402	4518	4521#		
EE13	023406	4520	4524#		
EE2	023224	4483#	4486		
EE3	023232	4482	4485#		
EE4	023240	4484	4488#	4559	
EE5	023266	4495#	4498		
EE6	023274	4494	4497#		
EE7	023302	4496	4500#	4563	
EE8	023332	4507#	4510		
EE9	023340	4506	4509#		
EMTVEC=	000030	149#	1268*	1269*	
EM1	070420	617	11742#		
EM10	071360	638	11833#		
EM100	100062	12491#			
EM103	100122	909	12497#		
EM104	100203	916	12506#		
EM105	100307	923	12518#		
EM106	100363	930	12526#		
EM107	100453	937	12536#		
EM11	071451	641	11845#		
EM110	100572	944	12550#		
EM111	100620	951	12554#		
EM112	100702	958	12563#		
EM113	100743	965	12569#		
EM115	101001	977	12575#		
EM12	071537	644	11856#		
EM123	101116	1008	12590#		
EM124	101164	1013	12597#		
EM125	101226	1019	12603#		
EM126	101266	1025	12609#		
EM127	101417	1032	12624#		
EM13	071664	647	11873#		
EM130	101546	1038	12640#		
EM131	101614	1044	12647#		
EM132	101656	1050	12653#		
EM133	101724	1056	12660#		
EM136	101766	1067	12666#		
EM137	102203	1070	12691#		
EM14	071744	650	11883#		
EM140	102421	1073	12717#		
EM141	102762	1076	12759#		
EM142	103322	1079	12801#		
EM143	103664	1082	12843#		
EM144	104225	1085	12885#		
EM145	104557	1088	12925#		
EM146	105110	1091	12966#		
EM147	105443	1094	13007#		
EM15	072003	653	11890#		
EM150	105775	1097	13047#		
EM151	106061	1100	13057#	13067	13068
EM152 =	106061	1103	13067#		
EM153 =	106061	1106	13068#		
EM154	106142	1109	13070#		
EM155	106174	1112	13076#		

EM156	106226	1115	13082#			
EM16	072053	656	11898#			
EM160	106273	1121	13091#			
EM161	106325	1124	13097#			
EM162	106373	1128	13104#			
EM163	106603	1132	13129#			
EM164	106675	1136	13140#			
EM165	106754	1140	13148#			
EM166	106773	1144	13151#			
EM167	107057	1148	13160#			
EM17	072127	659	11907#	11917		
EM170	107163	1152	13172#			
EM171	107226	1156	13178#			
EM172	107272	1160	13184#			
EM173	107356	1164	13194#			
EM174	107544	1168	13215#			
EM175	107607	1172	13221#			
EM2	070505	620	11752#			
EM20 =	072127	662	11917#			
EM21	072210	665	11919#			
EM22	072274	668	11929#	11973	12016	12036
EM23	072510	671	11955#	11975		
EM24 =	072274	674	11973#			
EM25 =	072510	677	11975#			
EM26	072644	680	11977#			
EM27	073011	683	11996#			
EM3	070677	623	11776#			
EM30 =	072274	686	12016#			
EM31	073156	689	12018#	12038		
EM32 -	072274	692	12036#			
EM33 -	073156	695	12038#			
EM34	073315	698	12040#			
EM35	073421	701	12053#			
EM36	073530	704	12067#			
EM37	073662	707	12084#			
EM4	071013	626	11790#			
EM40	073744	710	12094#			
EM41	074117	713	12115#			
EM42	074303	716	12137#			
EM43	074556	719	12169#			
EM435	107424	13201#				
EM44	074704	722	12185#	12264		
EM45	075100	725	12208#	12266		
EM46	075277	728	12232#	12268		
EM47	075420	731	12248#	12270		
EM5	071126	629	11804#			
EM50 =	074704	734	12264#			
EM51 =	075100	737	12266#			
EM52 =	075277	740	12268#			
EM53 =	075420	743	12270#			
EM54	075544	746	12272#			
EM55	075717	751	12291#			
EM56	075750	760	12296#			
EM57	076016	768	12303#			
EM6	071206	632	11813#			
EM60	076057	776	12309#			

PARCNT	054470	5326	5480	10028#					
PDMSG1	065536	4791	11441#						
PDMSG2	065714	4807	11461#						
PIRQ =	177772	44#	1546*	1549*	1555*	1563*	1566*	1572*	
PIRQVE=	000240	154#	1539*	1556*					
POWERM	065031	9751	11362#						
PP =	000011	2599#	2663						
PPHIAD	012250	2609#	2613						
PPLIM	012436	2612*	2615	2617*	2621	2636	2652	2656#	
PPLOAD	012246	2608#	2615	2617					
PP1	012236	2606#							
PP2	012304	2614	2616	2619#					
PP3	012324	2625#	2637						
PP4	012354	2627	2635#						
PP5	012370	2640#	2653						
PP6	012424	2644	2651#						
PP7	012440	2654	2658#						
PR0 =	000000	76#							
PR1 =	000040	77#							
PR2 =	000100	78#							
PR3 =	000140	79#							
PR4 =	000200	80#							
PR5 =	000240	81#	1576	1593					
PR6 =	000300	82#							
PR7 =	000340	83#	1540	1602	1622				
PS =	177776	41#	42	1260*	9050	9790			
PSW =	177776	42#	9952*	10597*	10757*	10916*	11102*	11231*	
PWRVEC=	000024	148#	1272*	1273*	9724*	9725*	9733*	9748*	9749*
QQERR1	032034	5704	5823#						
QQERR2	032112	5831	5834#						
QQERR3	032124	5835	5838#						
QQERR4	032126	5833	5837	5839#					
QQFLG1	031520	5639*	5661	5732#	5812*	5829			
QQFLG2	031516	5625*	5730#	5817*					
QQGM	031534	5632*	5645	5666	5738#	5810*			
QQGS	031532	5631*	5647	5653	5737#	5811*			
QQHI	031514	5677*	5686*	5695*	5698*	5719	5724	5727#	
QQLO	031512	5676*	5685*	5694*	5697*	5723	5726#		
QQPAT1	031522	5621*	5656	5733#	5816*	5834			
QQTMP1	031524	5719*	5720*	5721	5734#				
QQTMP2	031526	5735#							
QQTMP3	031530	5736#							
QQ1	031032	5629#	5819						
QQ10	031536	5670	5686	5688	5694	5698	5746#		
QQ11	031612	5672	5759#						
QQ12	031614	5676	5679	5690	5695	5697	5766#		
QQ13	031670	5681	5779#						
QQ14	031672	5677	5685	5786#					
QQ15	031746	5799#							
QQ16	031750	5755	5775	5795	5801#	5839			
QQ17	031752	5804#							
QQ18	032014	5813	5816#						
QQ19	032132	5818	5841#						
QQ2	031062	5643#	5814						
QQ3	031146	5656#	5665						
QQ4	031210	5660	5665#						

RK5DA2	060352	10902#	10924*							
RK5DB	004060	1239#								
RK5DD	040732	7336*	7367#							
RK5DFL	056053	10308*	10379#	10493*						
RK5DS	004044	1233#	10353	10980	10988	11049	11062			
RK5EE	040734	7342*	7368#							
RK5ER	004046	1234#	10979	11052	11061					
RK5ER1	060330	7446	10302*	10486	10893#	10930*	10981*	11063*		
RK5ER2	060332	7451	10894#	10931*	10979*	11061*				
RK5ER3	060334	7453	10895#	10932*	10980*	11062*				
RK5ER4	060336	7452	10896#							
RK5FF	040736	7349*	7369#							
RK5FLG	060326	10297*	10892#	10911	10954*	10964*	10969*			
RK5FT	042751	7819#								
RK5FUN	060344	10899#	10921*	10953						
RK5GG	040760	7371	7379#							
RK5HAN	060372	9717	10911#							
RK5HH	040773	7327*	7385#							
RK5H1	060404	10912	10916#							
RK5H2	060524	10944#								
RK5H3	060630	10959	10963#							
RK5H4	060644	10951	10968#							
RK5H5	060672	10963	10968	10975#						
RK5H51	060674	10976#	10987	10989						
RK5H6	060736	10978	10986#							
RK5II	040774	7334*	7386#							
RK5JJ	040776	7337*	7387#							
RK5KK	041000	7343*	7388#							
RK5LL	041002	7350*	7389#							
RK5MA1	060354	10903#	10925*	10947	11022	11033*				
RK5MA2	060356	10904#	10926*	10948	11023	11032*				
RK5MM	041024	7391	7398#							
RK5NN	041037	7328*	7405#							
RK5OO	041040	7335*	7406#							
RK5PP	041042	7339*	7407#							
RK5QQ	041044	7344*	7408#							
RK5RB	036134	6776#	7311	7312						
RK5RDY	061160	10935	11036#							
RK5REG	004042	1232#	10355							
RK5RR	041046	7346*	7409#							
RK5SEC	060366	10908#	11008*							
RK5SS	041132	7429	7435#							
RK5SUN	036110	6763#	7324							
RK5S1	060754	10934	10993#							
RK5S2	060776	10937	10999#							
RK5S3	061100	10940	11020#							
RK5TMP	060342	10898#	10917*	10920						
RK5TRK	060364	10907#	11009*							
RK5UNI	060346	10900#	10922*	10945	10993	10996*	11036	11041		
RK5USE	060340	10897#								
RK5V	004106	1254#	10950	10970*						
RK5VEC	060362	10906#	10928*	10972						
RK5WC	004052	1236#	10946*							
RK5WCT	060360	10905#	10927*	10946	11020*					
RK5XX	041150	7439	7442#							
RK5YY	041156	7374	7380	7394	7399	7414	7418	7431	7436	7445#

UBEBA	004070	1245#	11123*						
UBEBB	041432	7511#							
UBECC	004066	1244#	11122*						
UBECCC	041434	7497*	7512#						
JBECLR	004074	1247#	11181*	11197*					
UBECR	036100	6758#	6785	7540*	7550*				
UBECR1	004072	1246#	10498*	10499	11131*	11147	11151	11183	11190
UBECR2	004076	1248#	11124*	11152	11191				11198
UBECYL	061404	11095#							
UBEDA1	061364	11087#	11109*						
UBEDA2	061366	11088#	11110*	11125					
UBEDB	004064	1243#	10363	11125*					
UBEDD	041436	7481*	7513#						
UBEDFL	056054	10309*	10380#	10501*					
UBEEE	041440	7484*	7514#						
UBEER1	061344	7546	10303*	11079#	11115*	11153*			
UBEER2	061346	7551	11080#	11116*	11151*	11190*			
UBEER3	061350	7552	11081#	11117*	11152*	11191*	11192*		
UBEER4	061352	11082#							
UBEFF	041460	7516	7522#						
UBEFLG	061342	10298*	11078#	11097	11130*	11138*	11142*		
UBEFT	042752	7820#							
UBEFUN	061360	11085#	11108*	11129					
UBEGG	041472	7527#							
UBEHAN	061406	9718	11097#						
UBEHH	041474	7496*	7528#						
UBEH1	061420	11098	11102#						
UBEH2	061524	11122#							
UBEH3	061616	11133	11137#						
UBEH4	061632	11127	11142#						
UBEH5	061646	11137	11143	11146#					
UBEH51	061650	11147#	11159						
UBEH6	061712	11149	11158#						
UBEII	041476	7482*	7529#						
UBEJJ	041500	7485*	7530#						
UBEKK	041520	7532	7538#						
UBELL	041534	7541	7544#						
UBEMA1	061370	11089#	11111*	11123	11163	11176*			
UBEMA2	061372	11090#	11112*	11124	11164	11175*			
UBERB	036136	6777#	7475	7477					
UBERDY	062004	11118	11181#						
UBEREG	004062	1242#	10365						
UBESEC	061402	11094#							
UBESUN	036112	6764#							
UBES1	061722	11119	11163#						
UBETMP	061356	11084#	11103*	11106					
UBETRK	061400	11093#							
UBEUNI	061362	11086#							
UBEUSE	061354	11083#							
UBEV	004110	1255#	11126						
UBEVEC	061376	11092#	11114*	11144					
UBEWCT	061374	11091#	11113*	11122	11177*	11178*			
UBEYY	041540	7518	7522	7534	7538	7546#			
UBEZZ	041600	7547	7557#						
UCB	001000	431#	8618	8645					
UDPARO	177660	231#							

\$REG12	001602	553#													
\$REG13	001604	554#													
\$REG14	001606	555#													
\$REG15	001610	556#													
\$REG16	001612	557#													
\$REG17	001614	558#													
\$REG2	001562	545#	8148*	8184*	8256*	8264*	8351*	8387*	8465*	8511*	8631*	8657*	8716*	8731*	
		8819*	9122*	9143*	9175*	14016	14018	14021	14026						
\$REG20	001616	559#													
\$REG21	001620	560#													
\$REG22	001622	561#													
\$REG23	001624	562#													
\$REG3	001564	546#	8820*	9123*	9144*	9176*	14018	14021	14026						
\$REG4	001566	547#	9124*	9145*	9177*	14018	14026								
\$REG5	001570	548#													
\$REG6	001572	549#													
\$REG7	001574	550#													
\$RESRE	052064	9397#	9701												
\$SAVRE	052026	9381#	9700												
\$SAVR6	053340	9732*	9738	9739*	9740*	9757#									
\$SCOPE	051354	1266	9260#												
\$SETUP=	000037	451#	1266	1268	1270	1272	1274	1275	1276	1278	1282	1290	9214	9346	
\$SIZE	053342	1302	9781#												
\$SIZEX	053622	9820	9830#												
\$STUP -	177777	451#													
\$SVLAD	051572	9270	9302#												
\$SVPC -	000204	501#	506												
\$SWR -	167400	11	12#	13#	21	22	23	24	25	26	27	28	583	584	
		585	1275	1276	1278	1279	1441	1492	1532	1645	1763	1991	2207	2410	
		2598	2670	2802	3082	3322	3589	3885	4169	4302	4440	4597	4773	4838	
		5022	5138	5315	5469	5614	5867	6209	6665	7686	7927	7972	8002	8028	
		8057	8104	8218	8307	8418	8546	8591	8700	8749	8874	8942	8978	9013	
		9048	9209	9215	9230	9240	9241	9252	9253	9254	9255	9256	9261	9273	
		9275	9276	9282	9283	9284	9291	9292	9293	9304	9307	9310	9319	9320	
		9321	9322	9323	9324	9332	9335	9342	9346	9352	9362				
\$SWRMK -	000200	14#	28	29	9256	9257	9277	9279							
\$TAB	065076	10217	11370#	13876	13903	13913	13922								
\$TIMES	001676	583#	1275*	1763*	1991*	2207*	2410*	2598*	2670*	2802*	3082*	3322*	3589*	3885*	
		4169*	4302*	4440*	4597*	4838*	5022*	5138*	5315*	5469*	5867*	6209*	8942*	8978*	
		9013*	9215*	9291*	9298	9301*	9310								
\$TKB	001542	534#	1429*	10046	10071*										
\$TKS	001540	533#	1263	1430*	10072*										
\$TMP0	001626	563#	1768*	1996*	2212*	2415*	2603*	2675*	2806*	3088*	3327*	3596*	3892*	4174*	
		4307*	4445*	4602*	4777*	4843*	5027*	5143*	5320*	5474*	5619*	5872*	6214*	6669*	
		7690*	13855	13858	13866	13876	13881	13885	13888	13891	13893	13897	13907	13917	
		13926	13929	13932	13935	13938	13941	13944	13947	13950	13953	13956	13961	13964	
		13968	13973	13979	13988	13996	14005	14013	14030	14035	14052	14055	14060		
\$TMP1	001630	564#	1931*	2161*	2360*	2557*	2958*	2978*	3194*	3439*	4887*	4893*	4903*	4909*	
		4919*	4925*	4935*	4941*	5053*	5077*	5084*	5099*	5108*	5171*	5195*	5214*	5258*	
		5286*	5341*	5387*	5495*	5541*	5661*	5823*	5830	5927*	5942*	5956*	5971*	6030*	
		6050*	6069*	6089*	6184*	6255*	6270*	6284*	6299*	6330*	6347*	6366*	6383*	6402*	
		6419*	6438*	6455*	6482*	6505*	6528*	6551*	6573*	6577*	6586*	6591*	6950*	7118*	
		7282*	7451*	7551*	7713*	7753*	7776*	7799*	9898*	9915*	9928*	13855	13858	13876	
		13881	13885	13888	13891	13893	13897	13907	13917	13929	13935	13938	13941	13947	
		13953	13964	14052	14055	14060									
\$TMP10	001646	571#	3252*	3820*	4118*	4253*	4284*	4388*	4419*	4549*	4580*	4709*	4740*	5900*	

UMAC2	1#	5424	5434	5578	5588										
UMAC3	1#	5367	5413	5521	5567										
SSCMRE	508#	543	544	545	546	547	548	549	550	551	552	553	554	555	556
	557	558	559	560	561	562									
SSCMTM	508#	563	564	565	566	567	568	569	570	571	572	573	574	575	576
	577	578	579	580	581	582									
SSESCA	1#	421#													
SSNEWT	1#	421#	1434	1482	1524	1634	1704	1968	2193	2396	2588	2661	2779	3071	3284
	3564	3860	4156	4293	4427	4588	4749	4830	5014	5130	5297	5451	5606	5844	6194
	6606	7676	7919	7964	7994	8019	8049	8084	8202	8285	8405	8536	8570	8685	8742
	8859	8937	8973	9008	9043										
SSSET	9686#	9696	9697	9698	9699	9700	9701	9703	9704	9705	9706	9707	9708	9709	9710
	9711	9712	9714	9715	9716	9717	9718								
SSSKIP	1#	421#	7989	8198	8277	8401	8529	8677	8727	8736	8853	8946	8982	9017	
.EQUAT	1#														
.HEADE	1#														
.KT11	1#														
.SETUP	1#	451													
.SWRHI	1#	16													
.SWRLO	29#														
.SACT1	1#	481													
.SCATC	1#	468													
.SCMTA	1#	508													
\$.SDB2D	1#														
\$.SDB2O	1#	9854													
\$.SDIV	1#														
\$.SEOP	1#	9202													
\$.SERRO	1#	9312													
\$.SERRT	1#														
\$.SMULT	1#														
\$.SPOWE	1#	9719													
\$.SRAND	1#	9632													
\$.SRDDE	1#														
\$.SRDOC	1#														
\$.SREAD	1#														
\$.SSAVE	1#	9363													
\$.SSB2D	1#														
\$.SSB2O	1#														
\$.SSCOP	1#	9246													
\$.SSIZE	1#	9758													
\$.SSUPR	1#														
\$.STRAP	1#	9670													
\$.STYPB	1#														
\$.STYPD	1#	9563													
\$.STYPE	1#	9410													
\$.STYPO	1#	9484													
.1170	1#	31													

. ABS. 116712 000

ERRORS DETECTED: 0

DSKZ:CEKBDD.BIN,DSKZ:CEKBDD.LST/CRF/SOL CEKBDD.SML,CEKBDD.P11
 RUN-TIME: 80 120 18 SECONDS

RUN-TIME RATIO: 322/219=1.4
CORE USED: 38K (75 PAGES)