.REPT 0

IDENTIFICATION

Product Code:	MAINDEC-11-DQKKA-A-D
PRODUCT NAME:	11/6X CACHE DIAGNOSTIC
DATE:	MARCH,1977
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	WARREN SALTZ

TABLE OF CONTENTS

1.0 ABSTRACT

2.0 SYSTEM REQUIREMENTS

- 2.1 Hardware
- 2.2 Software
- 2.3 APT Setup
- 2.4 Execution Time

3.0 DIAGNOSTIC HIERARCHY PREREQUISITES

4.0 STARTING ADDRESS

5.0 PROGRAM CONTROL AND OPERATOR ACTION

6.0 SWITCH OPTIONS

7.0 PROGRAM DESCRIPTION

8.0 ERROR REPORTING AND FAULT ISOLATION

9.0 HANDLERS AND COMMON ROUTINES

- 9.1 End of Pass Routine
- 9.2 Scope Handler
- 9.3 Error Handler
- 9.4 Memory Size Routine
- 9.5 Trap Handler
- 9.6 Power Down and Up Routine
- 9.7 Trap Catcher
- 9.8 UPERR Routine
- 9.9 UT4 Routine
- 9.10 VIP Routine
- 9.11 TAG Routine
- 9.12 VEC Routine
- 9.13 HUBEN Routine
- 9.14 HUBEO Routine
- 9.15 HRK05 Routine
- 9.16 HRP03 Routine
- 9.17 HTU10 Routine
- 9.18 HAD Routine
- 9.19 Sweep Routine

1.0 ABSTRACT

The 11/6X Cache Diagnostic is comprised of a series of tests which were designed to check the cache's data paths on the Cache/KT board and its control logic on the Bus Control module. The tests are arranged in a logical order such that they build on one another. That is, the currently running test will depend on logic exercised by previous tests. Basic cache operations are exercised first followed by address and data functions. Those tests requiring extensive amounts of cache functioning are done near the end of the program. This testing procedure should provide a very effective degree of fault isolation.

2.0 SYSTEM REQUIREMENTS

2.1 Hardware

1. A working 11/6X CPU
2. A minimum of 13K to a max of 124K of memory. 124K is needed for complete check of TAG memory.
3. A console terminal (not mandatory under APT)
4. One of the following peripherals if NPR DATOs are to be tested (SW08=1).

   a. Unibus Exercisor (M7885)
   b. Bus Tester (old)
   c. RK05
   d. RP03
   e. TU10

5. When running under APT and either the NPR DATO tests (SW08=1) or the power up tests (SW07=1) are to be run, the diagnostic assumes a default peripheral of the Unibus Exercisor (M7885). In addition it assumes its data buffer address (BEDB) is 770000.

2.2 Software

This diagnostic will run under ACT/APT, XXDP and stand alone. When running under one of the various system testers, there should be no peripheral device doing any NPR DATO traffic on the bus (except those specifically chosen and under control of the diagnostic).

2.3 APT Setup

When running under APT and the NPR device tests or the power down tests are to be run, the APT software switch reg (switch 8 & 7 respectively) should be set (see sec. 6.0). The default APT device must be present when this is done (see

2.1.5).

2.4 Execution Time

For an error free, first run pass on a PDQ with core memory, it takes approximately 15 seconds.

3.0 DIAGNOSTIC HIERARCHY PREREQUISITES

It is assumed that CPU, memory, KT and stack limit are working properly for this program to give correct error reports. If not, their respective diagnostic should be run before the cache diagnostic. In addition, if one of the peripheral devices (see 2.1-4) is chosen, it is assumed to be error free. If not, further tests using the device are skipped.

4.0 STARTING ADDRESS

200 for normal startup

5.0 PROGRAM CONTROL AND OPERATOR ACTION

5.1 The standard diagnostic loading procedures are to be followed.

5.2 Load address 200

5.3 If the power up test is to be run set switch 07=1. If not running under APT after the test is started and the message "POWER MACHINE DOWN AND THEN UP" is typed, the machine should be powered down and up. The test will then continue. If running under APT & SW07=1, the program assumes the Unibus Exerciser is available. There is no type out when the exerciser is used in this manner.

5.4 If one of the peripheral devices is available (see 2.1-4) and the NPR DATO tests are to be done, set switch 8=1. Upon start of the program, the following beginning message will be typed (under APT message is not typed see sec. 6.8):

```
"TYPE WHICH DEVICE SHOULD BE USED:"

0 - [carriage return] - Unibus Exercisor (M7885)
1 - [carriage return] - Bus Tester Old
2 - [carriage return] - RK05
3 - [carriage return] - RP03
4 - [carriage return] - TU10
```

Before any device is chosen, it should be powered up and in the Ready state. The device should be write enabled and a scratch disk or tape should be mounted if the corresponding peripheral is used. The operator should then choose one of the devices and indicate his choice with a carriage return. If an incorrect entry is made (<0 or >4) the message "?INVALID ENTRY, TRY AGAIN" is typed. The program then waits for a correct value to be chosen. A rubout feature is provided to delete a typing error.

Depending upon the operator's choice, different information will have to be supplied by the user. The dialogue for each device is as follows:

a. 0 - Unibus Exercisor new

The following message is printed:

"TYPE THE UBE'S DATA BUFFER ADDRESS"

The operator should then supply the requested information. If the data is valid, the program proceeds to the first test. If there is no response to the address, the following message is printed:

```
"DEVICE DOES NOT RESPOND;
 REFERENCE TO IT TRAPS TO 4."

"?INVALID ENTRY, TRY AGAIN."
```

If the entry typed is not a valid data buffer address, the following message is printed:

"?INVALID ENTRY, TRY AGAIN"

In either case, the user should retype the correct data buffer address or restart the test and choose another device.

b. 1 - Unibus Exercisor old

No further operator action is needed if the device is present. If a reference to it times out, the following message is typed:

```
"DEVICE DOES NOT RESPOND
 REFERENCE TO IT TRAPS TO 4"
```

The program then retypes the beginning message and the user must choose another device.

c. 2 - RK05

If the RK05 is present, the following message is printed:

```
"WHICH DRIVE SHOULD BE USED?
 TYPE 0-7 <CARRIAGE RETURN>"
```

The user should then type the device number he wishes to use and indicate his choice with a carriage return. If a valid drive is chosen (>0,=0 or <8) the program proceeds to the first test. If it is invalid, the following message is typed:

```
"?INVALID ENTRY, TRY AGAIN"
```

The operator should then choose a correct drive number or restart the test and choose another device.

If a reference to an RK05 register times out, the RK05 is assumed not present or inoperable. In this case the following message is typed:

```
"DEVICE DOES NOT RESPOND
 REFERENCE TO IT TRAPS TO 4"
```

The program then retypes the beginning message and the user must choose another device.

d. 3 - RP03

If the RP03 is present the following message is printed:

```
"WHICH DRIVE SHOULD BE USED?
 TYPE 0-7 <CARRIAGE RETURN>"
```

The user should then type the drive number he wishes to use and indicate his choice with a carriage return. If a valid drive is chosen (>0,=0 or <8), the program proceeds to the first test. If it is invalid, the following message is typed:

```
"?INVALID ENTRY, TRY AGAIN"
```

The operator should then choose a correct drive number or restart the test and choose another device.

If a reference to an RP03 register times out, the RP03 is assumed not present or inoperable. In this case the following message is typed:

```
"DEVICE DOES NOT RESPOND
 REFERENCE TO IT TRAPS TO 4"
```

The program then retypes the beginning message and the user must choose another device.

e. 4 - TU10

If the TU10 is present, the following message is printed:

```
"WHICH DRIVE SHOULD BE USED?
 TYPE 0-7 <CARRIAGE RETURN>"
```

A scratch tape should be mounted and the user should then type the drive number he wishes to use and indicate his choice with a carriage return. If a valid drive number is chosen, the device is selected properly, and the write protect is off, the program proceeds to the first test. If any of the above are false the proper message is typed. The operator should then correct the problem and then choose another drive number.

If in the initial set up of the tape drive the ready bit fails to set or the error bit sets, one of the following messages is then typed:

```
"DEVICE READY BIT DOES NOT SET"

              or

"DEVICE ERROR BIT SET"
```

In either case the TU10 is assumed defective and the beginning message is then typed. The user must then choose another device.

5.5 Start the Program

6.0 SWITCH OPTIONS

```
SW<15>=1=100000  Halt on Error
SW<14>=1=040000  Loop on Test
SW<13>=1=020000  Inhibit Error Typeouts
SW<12>=1=010000  Inhibit Tests Using Memory Management
SW<11>=1=004000  Inhibit Iterations
SW<10>=1=002000  Bell on Error
SW<09>=1=001000  Loop on Error
SW<08>=1=000400  Enable NPR Device Tests
SW<07>=1=000200  Enable Power up Test
```

6.1 SW<15>

When set, the program halts on encountering an error after printing out the error message. Pressing continue restores normal program operation.

6.2 SW<14>

The program loops on the subtest that is being executed when the switch is set.

6.3 SW<13>

When set, this switch inhibits all error typeouts.

6.4 SW<12>

When set, this switch inhibits those tests using memory management. This switch should only be used when there is reason to believe that the KT is failing. Significant portions of cache will not be tested when this switch is set.

6.5 SW<11>

when set, iterations of each test is inhibited.

6.6 SW<10>

When set, the bell is rung upon encountering an error.

6.7 SW<09>

When set, upon finding an error, the program will cycle from the point of error to the previous scope statement or error loop ($LPERR). (see sec. 9.2).

6.8 SW<08>

When set, the NPR device tests will be run. It also enables the user interactive questions at the start of the test (see sec. 5.4). These questions are only asked on the first pass of the program. This switch should only be set before the program is started. When running under APT a default NPR device (Unibus Exercisor) is assumed and no questions are asked.

6.9 SW<07>

When set, the power up test is run (see sec. 5.3). This switch should not be set when running under ACT since user intervention is required. When running under APT a default device (Unibus Exercisor) is assumed.

7.0 PROGRAM DESCRIPTION

Upon start of the program, the cache is immediately turned off (force miss is on for both halfs of cache). The tests then proceed to selectively turn on only the half of cache that is to be exercised. The half of cache that is on is the half where the test locations reside. The half that is off always corresponds to the address space of the test instructions. This is to ensure that the instructions are not executed out of a possibly bad cache. In order to implement this scheme, the program was made non-contiguous between certain subtests.

The tests are structured on a half cache basis. That is several tests may be run on the low cache and then when the instruction address space has changed sufficiently to overlap the low cache addresses, the same tests will be repeated for the high cache addresses (low cache is defined as that portion of cache with physical address A10=0, high cache is defined as that portion of cache with physical address A10=1). This is done until cache is sufficiently checked out to assure that when all of it is turned on, there is a high probability that instructions can be executed out of it.

To facilitate the testing of cache, a 1K buffer is reserved at the end of the program for read and write operations. The starting address is BUFL corresponding to the first low cache address (A1-A9=0). The address BUFH corresponds to the first high cache address.

Immediately after the program is started the program identifies itself and then if SW8=1 it will interrogate the user about which peripheral device to use for the entire test (see sec. 5.4). This is only done on program start and not repeated for subsequent program loops. The interrogation is not done if running under APT. After this tests 1-47 are run.

8.0 ERROR REPORTING AND FAULT ISOLATION

Error calls are made via the EMT instruction. The lower byte of the instruction is encoded to indicate the error number. For example ERROR 1 would be (EMT+1) or 104001. Once an error instruction is executed, an error handler routine will then process the error call. The error message to be typed is determined from the item table at the end of the program. Item 1 corresponds to error 1 and so on. The item table contains a series of pointers to the message to be typed.

All error messages are identified by the words "ERROR: " or "FATAL ERROR: ".

A fatal error is a catastrophic failure which would cause all further printouts to be wrong or misleading. This is because fatal errors are only used to report failures in the hit reg and the cache control register. The entire diagnostic depends on this hardware functioning. A fatal error aborts the program and end of pass count is typed. In an "error" typeout only the individual test will be skipped. In some instances, the test will be continued until a max number of errors (usually 3) have been encountered. This is only done in cases where additional error information would aid in isolation.

The contents of the error reports identifies the hardware under test at the time of failure. Other pertinent information such as contents of cache control fields and failing addresses are also reported. The address information is reported as physical address high (P ADDH) corresponding to address bits A17, A16 and physical address low (P ADDL) corresponding to A15-A0.

When trouble shooting a failing board, the first error reported should be the first one fixed. This is because the nature of the software and hardware can create additional, false or misleading error messages to appear after the first one. Since the tests build on one another and involve previously tested hardware, it will aid in the fault isolation to look up the tests previously run to know which hardware has been tested. It should be pointed out that the probability of the error lying on the bus control board will decrease after the basic cache tests are successfully completed. The bus control contains a great deal of cache's hardcore control logic which if not functioning will mean, many times, that the cache diagnostic or any program can not run out of cache. Because of this, if the diagnostic reports an error, there is a higher probability of it lying on the Cache/KT board than the Bus Control board.

9.0 HANDLERS AND COMMON ROUTINES

9.1 End of Pass Routine

This routine takes care of transferring control to the monitor (if one exists) or to the beginning of the program. It indicates the pass number each time it is executed.

9.2 Scope Handler

This handler is called via the 'IOT' trap. When 'scope' is executed an 'IOT' trap occurs to the memory location '$SCOPE'. Depending on the switch settings, the handler then decides to loop on test, loop on error etc. The scope statement that is located at the first instruction of the following test is the one that enabled the desired action (looping etc.) for the present test.

9.3 Error Handler

This handler uses the 'EMT' trap. The lower byte of the instruction is encoded to indicate the error number. For example ERROR 1 would be (EMT+1) or 104001. Once an error instruction is executed the error handler determines the message to be typed. An item table at the end of the program contains pointers for each message to be typed. Each item corresponds to each error (Item 1 corresponds to error 1). The 'ERRTYP' routine then processes the table for the final error type out.

9.4 Memory Size Routine

This routine sizes memory to find the maximum memory size. If bit7 of location $KT11=1, before the routine is called, memory management will be used. $LSTAD contains the last virtual address of the last bank if memory management is used. Otherwise it contains the last absolute address of available memory. $LSTBK will contain the last bank as a page address register.

9.5 Trap Handler

This handler uses the trap instruction. The lower byte of the instruction is encoded differently for each of the different routines that use it. When a call for a routine is executed a trap occurs to the handler located at $TRAP. The handler then determines by looking at the lower byte which address to go to for servicing the call. The following routines use this handler:

1. TYPE - this routine is used to type ASCIZ messages.
2. TYPOCT, TYPOS & TYPON - These routines are used to change a binary number to a 6 digit octal number and type it.
3. RDOCT - this routine will read an octal number from the TTY.
4. RDLIN - this routine will input an ASCII string from the TTY.
5. TYPDS - this routine converts a binary number to decimal and types it.

9.6 Power Down and Up Routines

When a power fail condition occurs, the contents of registers R0-R7 are saved on the stack. When the power returns, the same registers are restored.

9.7 Trap Catcher

This is a series of instructions starting in location 0 to detect unexpected traps and interrupts to the trap and interrupt vector area of memory.

Each vector PC address is loaded with the address of the next location. The next location is loaded with a halt. Thus an illegal trap or interrupt will cause a halt at the trap PSW location plus 2.

Once a halt occurs, by examining the contents of the address pointed to by the stack, the value of the PC when the trap or interrupt occurred can be determined.

9.8 UPERR

This subroutine is used to report unexpected parity errors while the program is running. At the beginning of each test a pointer to the next test is saved. Any spurious parity error is reported and then the test following the one with the error is started.

9.9 UT4

This subroutine reports unexpected traps to 4. After the error is reported, the machine will be halted. Pressing continue will restart the program.

9.10 VIP

This subroutine takes a virtual address stored in location $TMP0 and converts it to a physical address. The physical address bits A17, A16 are stored in $REG1 and bits A0 - A15 are stored in $REG2.

9.11 TAG

This subroutine calculates the tag field from a page address register's contents stored in $TMP0.

9.12 VEC

This subroutine finds out if a new Unibus Exercisor module is being used and if so puts an RTI in its interrupt vector.

9.13 HUBEN

This subroutine sets up the new Unibus exercisor to do one NPR DATO to the address following the subroutine call.

9.14 HUBEO

This subroutine sets up the old Unibus Exercisor to do one NPR DATO to the address following the subroutine call.

9.15 HRK05

This subroutine sets up the RK05 to do NPR DATO's to the starting address following the subroutine call.

9.16 HRP03

This subroutine sets up the RP03 to do NPR DATO's to the starting address following the subroutine call.

9.17 HTU10

This subroutine sets up the TU10 to do NPR DATO's to the starting address following the subroutine call.

9.18 HAD

This subroutine generates an address in a 1K test buffer at the end of the program. The address is (512)10 locations from the given address following this subroutine call.

9.19 SWEEP

This routine rids cache of bad parity. It is called after all cache has been turned off.

.ENDR

```
   15          OPERATIONAL SWITCH SETTINGS
   31          BASIC DEFINITIONS
  141          MEMORY MANAGEMENT DEFINITIONS
  340          TRAP CATCHER
  353          STARTING ADDRESS(S)
  360          APT PARAMETER BLOCK
  382          ACT11 HOOKS
  396          COMMON TAGS
  457          APT MAILBOX-ETABLE
  575          INITIALIZE THE COMMON TAGS
  882          T1      TEST PA MUX AND PHYSICAL ADDRESS DRIVERS
 1092          T2      TEST CACHE CAN BE TURNED OFF AND HIT REG CLEARED
 1133          T3      TEST CAN GET A HIT ON A HIGH CACHE ADDRESS AND HIT REG CAN =1
 1205          T4      TEST FORCE MISS ON HIGH ADDRESS
 1229          T5      TEST CACHE TRACKS WHEN CACHE IS OFF
 1263          T6      TEST DATOB OPERATION
 1323          T7      TEST DATO ALLOCATES CACHE
 1362          T10     TEST CAN GET HIT AND  FORCE MISS ON LOW CACHE ADDRESS
 1409          T11     TEST OF TAG ADDRESS COMPARATOR
 1511          T12     TEST FORCE MISS LOCKS OUT PARITY ERRORS & CCR WWP CAN =1
 1609          T13     TEST OF TAG PARITY GENERATOR/CHECKER
 1824          T14     TEST OF DATA PARITY GENERATOR/CHECKER
 2007          T15     TEST THE VALID BIT FOR LOW HALF OF CACHE
 2213          T16     TEST TAG PARITY BIT FOR LOW CACHE ADDRESSES
 2354          T17     TEST DATA PARITY BITS FOR LOW CACHE
 2494          T20     TEST THE VALID BIT FOR HIGH HALF OF CACHE
 2686          T21     TEST TAG PARITY BIT FOR HIGH CACHE ADDRESSES
 2832          T22     TEST TAG ADDRESS BITS FOR LOW HALF OF CACHE
 3018          T23     TEST OF CACHE DATA LOC WITH FLOAT 1 & 0 PATTERNS
 3145          T24     TEST DATA PARITY BITS FOR HIGH CACHE
 3280          T25     TEST TAG ADDRESS BITS FOR HIGH HALF OF CACHE
 3472          T26     TEST DATA FIELD FOR LOW HALF OF CACHE
 3643          T27     TEST DATA FIELD FOR HIGH HALF OF CACHE
 3807          T30     TEST OF MSB ADDRESS (A10) TO VALID BIT
 3943          T31     TEST OF MSB ADDRESS (A10) TO CACHE TAG FIELD
 4092          T32     TEST OF MSB ADDRESS (A10) TO CACHE DATA FIELD
 4233          T33     TEST CACHE IS NOT ALLOCATED DURING ODD ADDRESS TRAP
 4281          T34     TEST CACHE NOT ALLOCATED DURING RED ZONE TRAP
 4332          T35     TEST CACHE NOT ALLOCATED DURING KT ABORT
 4395          T36     DYNAMIC TEST OF CACHE
 4612          T37     TEST RETRIES TO BACKING STORE DONE ON CACHE PARITY TRAP
 4679          T40     TEST DATO TO I/0 LOC NOT WRITTEN IN CACHE AND I/0
 4718          T41     TEST CONSOLE INITIATED SWEEP INVALIDATES ALL CACHE
 4782          T42     TEST POWER UP INVALIDATES CACHE AND CLEARS CACHE CONTROL REG
 4918          T43     TEST NPR DATO INVALIDATES CACHE FOR PHYSICAL ADDRESS BITS A1-A10
 5049          T44     TEST NPR DATO INVALIDATES CACHE FOR PHYSICAL ADDRESS BITS A17-A11
 5187          END OF PASS ROUTINE
 5590          SCOPE HANDLER ROUTINE
 5651          ERROR HANDLER ROUTINE
 5707          ERROR MESSAGE TYPEOUT ROUTINE
 5754                  ROUTINE TO SIZE MEMORY
 5846          CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 5913          TYPE ROUTINE
 5992          APT COMMUNICATIONS ROUTINE
 6049          BINARY TO OCTAL (ASCII) AND TYPE
 6126          TTY INPUT ROUTINE
```

```
  6228          READ AN OCTAL NUMBER FROM THE TTY
  6281          TRAP DECODER
  6304          TRAP TABLE
  6322          POWER DOWN AND UP ROUTINES
  7500          ERROR POINTER TABLE
```

```

                            .TITLE  MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC
                            ;*COPYRIGHT (C) APRIL 11,1975
                            ;*DIGITAL EQUIPMENT CORP.
                            ;*MAYNARD, MASS. 01754
                            ;*
                            ;*PROGRAM BY WARREN L. SALTZ
                            ;*
                            ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                            ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                            ;*
            000001          $TN=1
                            .SBTTL  OPERATIONAL SWITCH SETTINGS
                            ;*
                            ;*      SWITCH                  USE
                            ;*      ------          --------------------
                            ;*        15            HALT ON ERROR
                            ;*        14            LOOP ON TEST
                            ;*        13            INHIBIT ERROR TYPEOUTS
                            ;*        12            INHIBIT TEST USING MEMORY MANAGEMENT
                            ;*        11            INHIBIT ITERATIONS
                            ;*        10            BELL ON ERROR
                            ;*         9            LOOP ON ERROR
                            ;*         8            ENABLE NPR DEVICE TESTS
                            ;*         7            ENABLE POWER UP TEST

                            ;;*************************************************************
                            ;;*************************************************************
                            .SBTTL  BASIC DEFINITIONS

                            ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
            001100          STACK=  1100
                            .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                            .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                            ;*MISCELLANEOUS DEFINITIONS
            000011          HT=     11              ;;CODE FOR HORIZONTAL TAB
            000012          LF=     12              ;;CODE FOR LINE FEED
            000015          CR=     15              ;;CODE FOR CARRIAGE RETURN
            000200          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
            177776          PS=     177776          ;;PROCESSOR STATUS WORD
                            .EQUIV  PS,PSW
            177774          STKLMT= 177774          ;;STACK LIMIT REGISTER
            177772          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
            177570          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
            177570          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                            ;*GENERAL PURPOSE REGISTER DEFINITIONS
            000000          R0=     %0              ;;GENERAL REGISTER
            000001          R1=     %1              ;;GENERAL REGISTER
            000002          R2=     %2              ;;GENERAL REGISTER
            000003          R3=     %3              ;;GENERAL REGISTER
            000004          R4=     %4              ;;GENERAL REGISTER
            000005          R5=     %5              ;;GENERAL REGISTER
            000006          R6=     %6              ;;GENERAL REGISTER
            000007          R7=     %7              ;;GENERAL REGISTER
```

```
            000006          SP=     %6              ;;STACK POINTER
            000007          PC=     %7              ;;PROGRAM COUNTER

                            ;*PRIORITY LEVEL DEFINITIONS
            000000          PR0=    0               ;;PRIORITY LEVEL 0
            000040          PR1=    40              ;;PRIORITY LEVEL 1
            000100          PR2=    100             ;;PRIORITY LEVEL 2
            000140          PR3=    140             ;;PRIORITY LEVEL 3
            000200          PR4=    200             ;;PRIORITY LEVEL 4
            000240          PR5=    240             ;;PRIORITY LEVEL 5
            000300          PR6=    300             ;;PRIORITY LEVEL 6
            000340          PR7=    340             ;;PRIORITY LEVEL 7

                            ;*"SWITCH REGISTER" SWITCH DEFINITIONS
            100000          SW15=   100000
            040000          SW14=   40000

            020000          SW13=   20000
            010000          SW12=   10000
            004000          SW11=   4000
            002000          SW10=   2000
            001000          SW09=   1000
            000400          SW08=   400
            000200          SW07=   200
            000100          SW06=   100
            000040          SW05=   40
            000020          SW04=   20
            000010          SW03=   10
            000004          SW02=   4
            000002          SW01=   2
            000001          SW00=   1
                            .EQUIV  SW09,SW9
                            .EQUIV  SW08,SW8
                            .EQUIV  SW07,SW7
                            .EQUIV  SW06,SW6
                            .EQUIV  SW05,SW5
                            .EQUIV  SW04,SW4
                            .EQUIV  SW03,SW3
                            .EQUIV  SW02,SW2
                            .EQUIV  SW01,SW1
                            .EQUIV  SW00,SW0

                            ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
            100000          BIT15=  100000
            040000          BIT14=  40000
            020000          BIT13=  20000
            010000          BIT12=  10000
            004000          BIT11=  4000
            002000          BIT10=  2000
            001000          BIT09=  1000
            000400          BIT08=  400
            000200          BIT07=  200
            000100          BIT06=  100
            000040          BIT05=  40
            000020          BIT04=  20
            000010          BIT03=  10
            000004          BIT02=  4
```

```
        000002          BIT01= 2
        000001          BIT00= 1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
        000004          ERRVEC= 4                ;;TIME OUT AND OTHER ERRORS
        000010          RESVEC= 10               ;;RESERVED AND ILLEGAL INSTRUCTIONS
        000014          TBITVEC=14               ;;"T" BIT
        000014          TRTVEC= 14               ;;TRACE TRAP
        000014          BPTVEC= 14               ;;BREAKPOINT TRAP (BPT)
        000020          IOTVEC= 20               ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
        000024          PWRVEC= 24               ;;POWER FAIL
        000030          EMTVEC= 30               ;;EMULATOR TRAP (EMT) **ERROR**
        000034          TRAPVEC=34               ;;"TRAP" TRAP
        000060          TKVEC=  60               ;;TTY KEYBOARD VECTOR
        000064          TPVEC=  64               ;;TTY PRINTER VECTOR
        000240          PIRQVEC=240              ;;PROGRAM INTERRUPT REQUEST VECTOR
                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                        ;*KT11 VECTOR ADDRESS

        000250          MMVEC=  250

                        ;*KT11 STATUS REGISTER ADDRESSES

        177572          SR0=    177572
        177574          SR1=    177574
        177576          SR2=    177576
        172516          SR3=    172516

                        ;*USER "I" PAGE DESCRIPTOR REGISTERS

        177600          UIPDR0= 177600
        177602          UIPDR1= 177602
        177604          UIPDR2= 177604
        177606          UIPDR3= 177606
        177610          UIPDR4= 177610
        177612          UIPDR5= 177612
        177614          UIPDR6= 177614
        177616          UIPDR7= 177616

                        ;*USER "D" PAGE DESCRIPTOR REGISTORS

        177620          UDPDR0= 177620
        177622          UDPDR1= 177622
        177624          UDPDR2= 177624
        177626          UDPDR3= 177626
```

```
        177630          UDPDR4= 177630
        177632          UDPDR5= 177632
        177634          UDPDR6= 177634
        177636          UDPDR7= 177636

                        ;*USER "I" PAGE ADDRESS REGISTERS

        177640          UIPAR0= 177640
        177642          UIPAR1= 177642
        177644          UIPAR2= 177644
        177646          UIPAR3= 177646
        177650          UIPAR4= 177650
        177652          UIPAR5= 177652
        177654          UIPAR6= 177654
        177656          UIPAR7= 177656

                        ;*USER "D" PAGE ADDRESS REGISTERS

        177660          UDPAR0= 177660
        177662          UDPAR1= 177662
        177664          UDPAR2= 177664
        177666          UDPAR3= 177666
        177670          UDPAR4= 177670
        177672          UDPAR5= 177672
        177674          UDPAR6= 177674
        177676          UDPAR7= 177676

                        ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS

        172200          SIPDR0= 172200
        172202          SIPDR1= 172202
        172204          SIPDR2= 172204
        172206          SIPDR3= 172206
        172210          SIPDR4= 172210
        172212          SIPDR5= 172212
        172214          SIPDR6= 172214
        172216          SIPDR7= 172216

                        ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS

        172220          SDPDR0= 172220
        172222          SDPDR1= 172222
        172224          SDPDR2= 172224
        172226          SDPDR3= 172226
        172230          SDPDR4= 172230
        172232          SDPDR5= 172232
        172234          SDPDR6= 172234
        172236          SDPDR7= 172236

                        ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS

        172240          SIPAR0= 172240
        172242          SIPAR1= 172242
        172244          SIPAR2= 172244
        172246          SIPAR3= 172246
        172250          SIPAR4= 172250
        172252              SIPAR5= 172252
        172254              SIPAR6= 172254
        172256              SIPAR7= 172256

                            ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS

        172260              SDPAR0= 172260
        172262              SDPAR1= 172262
        172264              SDPAR2= 172264
        172266              SDPAR3= 172266
        172270              SDPAR4= 172270
        172272              SDPAR5= 172272
        172274              SDPAR6= 172274
        172276              SDPAR7= 172276

                            ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

        172300              KIPDR0= 172300
        172302              KIPDR1= 172302
        172304              KIPDR2= 172304
        172306              KIPDR3= 172306
        172310              KIPDR4= 172310
        172312              KIPDR5= 172312
        172314              KIPDR6= 172314
        172316              KIPDR7= 172316

                            ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS

        172320              KDPDR0= 172320
        172322              KDPDR1= 172322
        172324              KDPDR2= 172324
        172326              KDPDR3= 172326
        172330              KDPDR4= 172330
        172332              KDPDR5= 172332
        172334              KDPDR6= 172334
        172336              KDPDR7= 172336

                            ;*KERNEL "I" PAGE ADDRESS REGISTERS

        172340              KIPAR0= 172340
        172342              KIPAR1= 172342
        172344              KIPAR2= 172344
        172346              KIPAR3= 172346
        172350              KIPAR4= 172350
        172352              KIPAR5= 172352
        172354              KIPAR6= 172354
        172356              KIPAR7= 172356

                            ;*KERNEL "D" PAGE ADDRESS REGISTERS

        172360              KDPAR0= 172360
        172362              KDPAR1= 172362
        172364              KDPAR2= 172364
        172366              KDPAR3= 172366
        172370              KDPAR4= 172370
        172372              KDPAR5= 172372
```

```
        172374              KDPAR6= 172374
        172376              KDPAR7= 172376

                                          ;*OTHER EQUATES
        177752              HMR=177752                        ;HIT/MISS REG ADDRESS
        177746              CCR=177746                        ;CACHE CONTROL REG ADDRESS
        000106              CDH=106                                   ;CACHE DATA HIGH ADDRESS
        000106              CDL=106                                   ;CACHE DATA LOW ADDRESS
        000107              CTAG=107                                  ;CACHE TAG ADDRESS
        177744              EREG=177744                               ;MEMORY ERROR REG ADDRESS
        177766              CER=177766                        ;CPU ERROR REG ADDRESS
        000101              HIADD=101                         ;HIGH UNIBUS ADDRESS OF ERROR
        000102              LOADD=102                         ;LOW UNIBUS ADDRESS OF ERROR
        055016              BSD=55016                                 ;BACKING STORE DATA ADDRESS
        060000              BUFL=60000                        ;LOW ADDRESS BUFFER (A10=0)
        062000              BUFH=BUFL+2000                    ;HIGH ADDRESS BUFFER (A10=1)
        076600              MED=     76600                    ;MAINTENANCE INSTRUCTION
        000100              RJAM=    100                      ;LOG READ ADDRESS FOR JAM REG.
        000101              RSER=    101                      ;LOG READ ADDRESS FOR SERVICE REG.
        000102              RPBA=    102                      ;LOG READ ADDRESS FOR PHYSICAL BUS ADDR.
        000107              RTAG=    107                      ;LOG READ ADDRESS FOR CACHE TAG
        000106              RDAT=    106                      ;LOG READ ADDRESS FOR CACHE DATA
        000022              RLOG=    22                       ;READ ADDRESS FOR CPU INTERNAL REG "WHAMI"
        000222              WLOG=    222                      ; WRITE ADDRESS FOR CPU INTERNAL REG "WHAMI"
        000304              WFLI=    304                      ;WRITE ADDRESS FOR CPU INTERNAL REG "FLAG/INT"
        000226              WSW=     226                      ;WRITE ADDRESS FOR CPU INTERNAL REG "SWITCH REG"
        000352              WINIT=   352                      ;WRITE ADDRESS FOR CPU INTERNAL REG "INIT REG"(MOD FOR D
        177572              MMR0=SR0                          ;KT11 STATUS REG
        177576              MMR2=SR2                          ;KT11 STATUS REG
        000114              PVEC=114                          ;PARITY TRAP VECTOR
        177400              RKDS=    177400                   ;RK05 DRIVE STATUS REG
        177402              RKER=    177402                   ;RK05 ERROR REG
        177404              RKCS=    177404                   ;RK05 CONTROL STATUS REG
        177406              RKWC=    177406                   ;RK05 WORD COUNT REG
        177410              RKBA=    177410                   ;RK05 CURRENT BUS ADDRESS REG
        177412              RKDA=    177412                   ;RK05 DISK ADDRESS REG
        176710              RPDS=    176710                   ;RP03 DEVICE STATUS REG
        176712              RPER=    176712                   ;RP03 ERROR REG
        176714              RPCS=    176714                   ;RP03 CONTROL STATUS REG
        176716              RPWC=    176716                   ;RP03 WORD COUNT REG
        176720              RPBA=    176720                   ;RP03 BUS ADDRESS REG
        176722              RPCA=    176722                   ;RP03 CYLINDER ADDRESS REG
        176724              RPDA=    176724                   ;RP03 DISK ADDRESS REG
        172520              MTS=     172520                   ;TU10 STATUS REG
        172522              MTC=     172522                   ;TU10 COMMAND REG
        172524              MTBRC=   172524                   ;TU10 BYTE RECORD COUNTER
        172526              MTCMA=   172526                   ;TU10 CURRENT MEMORY ADDRESS REG
        000001              HMR0=    1                        ;HIT MISS REG BIT 0
        000002              HMR1=    2                        ;HIT MISS REG BIT 1
        000004              HMR2=    4                        ;HIT MISS REG BIT 2
        000010              HMR3=    10                       ;HIT MISS REG BIT 3
        000020              HMR4=    20                       ;HIT MISS REG BIT 4
        000040              HMR5=    40                       ;HIT MISS REG BIT 5
        000015              CR=      15                       ;CARRIAGE RETURN
        000012              LF=      12                       ;LINE FEED
```

```
                                    ;;*************************************************************
                                    .SBTTL  TRAP CATCHER

                 000000                     .=0
                                    ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                    ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                    ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                 000174                     .=174
        000174   000000             DISPREG: .WORD  0                   ;;SOFTWARE DISPLAY REGISTER
        000176   000000             SWREG:   .WORD  0                   ;;SOFTWARE SWITCH REGISTER
                                    ;;*************************************************************
                 000200                     LOC=.
                 000200                     .=200

                                    .SBTTL  STARTING ADDRESS(S)


        000200   012737  000214  177746         MOV     #214,@#CCR      ;TURN CACHE OFF
        000206   000137  001362                 JMP     @#START         ;JUMP TO STARTING ADDRESS OF PROGRAM.
                 000200                     .=LOC
                 001000                     .=1000
                                    .SBTTL  APT PARAMETER BLOCK

                                    ;;*************************************************************
                                    ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                    ;;*************************************************************
                 001000                     .$X=.   ;;SAVE CURRENT LOCATION
                 000024                     .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
        000024   000200                     200     ;;FOR APT START UP
                 000044                     .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
        000044   001000                     $APTHDR ;;POINT TO APT HEADER BLOCK
                 001000                     .=.$X   ;;RESET LOCATION COUNTER
                                    ;;*************************************************************
                                    ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                    ;INTERFACE SPEC.

        001000                      $APTHD:
        001000   000000             $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
        001002   001236             $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
        001004   000060             $TSTM:  .WORD   60      ;;RUN TIM OF LONGEST TEST
        001006   000060             $PASTM: .WORD   60      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
        001010   000000             $UNITM: .WORD           ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
        001012   000052                     .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
                                    .SBTTL  ACT11 HOOKS

                                    ;;*************************************************************
                                    ;HOOKS REQUIRED BY ACT11
                 001014                     $SVPC=.                     ;SAVE PC
                 000046                     .=46
        000046   033106                     $ENDAD                      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                 000052                     .=52
        000052   000000                     .WORD   0                   ;;2)SET LOC.52 TO ZERO
                 001014                     .=$SVPC                     ;; RESTORE PC


                                    ;*************************************************************
```

```

                                    .SBTTL  COMMON TAGS

                                    ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                    ;*USED IN THE PROGRAM.

                 001100                     .=1100
        001100                      $CMTAG:                             ;;START OF COMMON TAGS
        001100   000000                     .WORD   0
        001102      000             $TSTNM: .BYTE   0                   ;;CONTAINS THE TEST NUMBER
        001103      000             $ERFLG: .BYTE   0                   ;;CONTAINS ERROR FLAG
        001104   000000             $ICNT:  .WORD   0                   ;;CONTAINS SUBTEST ITERATION COUNT
        001106   000000             $LPADR: .WORD   0                   ;;CONTAINS SCOPE LOOP
        001110   000000             $LPERR: .WORD   0                   ;;CONTAINS SCOPE RETURN FOR ERRORS
        001112   000000             $ERTTL: .WORD   0                   ;;CONTAINS TOTAL ERRORS DETECTED
        001114      000             $ITEMB: .BYTE   0                   ;;CONTAINS ITEM CONTROL BYTE
        001115      001             $ERMAX: .BYTE   1                   ;;CONTAINS MAX. ERRORS PER TEST
        001116   000000             $ERRPC: .WORD   0                   ;;CONTAINS PC OF LAST ERROR INSTRUCTION
        001120   000000             $GDADR: .WORD   0                   ;;CONTAINS OF 'GOOD' DATA
        001122   000000             $BDADR: .WORD   0                   ;;CONTAINS OF 'BAD' DATA
        001124   000000             $GDDAT: .WORD   0                   ;;CONTAINS 'GOOD' DATA
        001126   000000             $BDDAT: .WORD   0                   ;;CONTAINS 'BAD' DATA
        001130   000000                     .WORD   0                   ;;RESERVED--NOT TO BE USED
        001132   000000                     .WORD   0
        001134   177570             SWR:    .WORD   DSWR                ;;OF SWITCH REGISTER
        001136   177570             DISPLAY:.WORD   DDISP               ;;OF DISPLAY REGISTER
        001140   177560             $TKS:   177560                      ;;TTY KBD STATUS
        001142   177562             $TKB:   177562                      ;;TTY KBD BUFFER
        001144   177564             $TPS:   177564                      ;;TTY PRINTER STATUS REG.
        001146   177566             $TPB:   177566                      ;;TTY PRINTER BUFFER REG.
        001150      000             $NULL:  .BYTE   0                   ;;CONTAINS NULL CHARACTER FOR FILLS
        001151      002             $FILLS: .BYTE   2                   ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
        001152      012             $FILLC: .BYTE   12                  ;;INSERT FILL CHARS. AFTER A "LINE FEED"
        001153      000             $TPFLG: .BYTE   0                   ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
        001154   000000             $REGAD: .WORD   0                   ;;CONTAINS THE FROM
                                                                        ;;WHICH ($REG0) WAS OBTAINED
        001156   000000             $REG0:  .WORD   0                   ;;CONTAINS (($REGAD)+0)
        001160   000000             $REG1:  .WORD   0                   ;;CONTAINS(($REGAD)+2)
        001162   000000             $REG2:  .WORD   0                   ;;CONTAINS(($REGAD)+4)
        001164   000000             $REG3:  .WORD   0                   ;;CONTAINS(($REGAD)+6)
        001166   000000             $REG4:  .WORD   0                   ;;CONTAINS(($REGAD)+10)
        001170   000000             $REG5:  .WORD   0                   ;;CONTAINS(($REGAD+12)
        001172   000000             $TMP0:  .WORD   0                   ;;USER DEFINED
        001174   000000             $TMP1:  .WORD                       ;;USER DEFINED
        001176   000000             $TMP2:  .WORD   0                   ;;USER DEFINED
        001200   000000             $TMP3:  .WORD   0                   ;;USER DEFINED
        001202   000000             $TMP4:  .WORD   0                   ;;USER DEFINED
        001204   000000             $TMP5:  .WORD   0                   ;;USER DEFINED
        001206      077             $QUES:  .ASCII  /?/                 ;;QUESTION MARK
        001207      015             $CRLF:  .ASCII  <15>                ;;CARRIAGE RETURN
        001210   000012             $LF:    .ASCIZ  <12>                ;;LINE FEED
        001212   000000             CREG1:  .WORD   0                   ;CONTROL REG ADDR. FOR NPR DEVICE
        001214   000000             CREG2:  .WORD   0                   ;CONTROL REG ADDR. FOR NPR DEVICE
        001216   000000             CREG3:  .WORD   0                   ;CONTROL REG ADDR. FOR NPR DEVICE
        001220   000000             CREG4:  .WORD   0                   ;CONTROL REG ADDR. FOR NPR DEVICE
        001222   000000             CREG5:  .WORD   0                   ;CONTROL REG ADDR. FOR NPR DEVICE
```

```
001224  000000          CREG6:  .WORD   0                       ;CONTROL REG ADDR. FOR NPR DEVICE
001226  000000          IVEC:   .WORD   0                       ;ADDRESS OF DEVICE'S INTERRUPT VECTOR
001230  000000          EAD:    .WORD   0                       ;ADDRESS OF DEVICE'S ERROR REG
001232  000000          SETUP:  .WORD   0                       ;ADDRESS OF DEVICE'S HANDLER
001234  000000          SKTST:  .WORD   0                       ;;POINTER TO TEST FOLLOWING ONE BEING EXECUTED

                        .SBTTL  APT MAILBOX-ETABLE

                        ;;*****************************************************************
                        .EVEN
001236                  $MAIL:                          ;;APT MAILBOX
001236  000000          $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001240  000000          $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001242  000000          $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001244  000000          $PASS:  .WORD   APASS   ;;PASS COUNT
001246  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001250  000000          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001252  000000          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001254  000000          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001256                  $ETABLE:                        ;;APT ENVIRONMENT TABLE
001256     000          $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001257     000          $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001260  000000          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001262  000000          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001264  000000          $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                        ;*                      BITS 15-11=CPU TYPE
                        ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                        ;*                              11/70=06,PDQ=07,Q=10
                        ;*                      BIT 10=REAL TIME CLOCK
                        ;*                      BIT  9=FLOATING POINT PROCESSOR
                        ;*                      BIT  8=MEMORY MANAGEMENT
001266     000          $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001267     000          $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                        ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                        ;*                              900 NSEC CORE=001
                        ;*                              300 NSEC BIPOLAR=002
                        ;*                              500 NSEC MOS=003
001270  000000          $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                        ;*                       MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001272     000          $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001273     000          $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001274  000000          $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001276     000          $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001277     000          $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001300  000000          $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001302     000          $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001303     000          $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001304  000000          $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001306  000000          $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001310  000000          $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001312  000000          $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001314  000000          $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001316  000000          $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001320  000000          $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001322  000000          $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001324  000000          $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
```

```
001326  000000          $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
001330  000000          $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001332  000000          $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
001334  000000          $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
001336  000000          $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001340  000000          $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001342  000000          $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
001344  000000          $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
001346  000000          $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
001350  000000          $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
001352  000000          $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
001354  000000          $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
001356  000000          $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
001360  000000          $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15


001362                  $ETEND:
```

```
                                      ;;*************************************************************
                                      ;;*************************************************************

001362                                START:
                                      .SBTTL  INITIALIZE THE COMMON TAGS
                                      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001362  012706  001100                        MOV     #$CMTAG,R6        ;;FIRST LOCATION TO BE CLEARED
001366  005026                                CLR     (R6)+             ;;CLEAR MEMORY LOCATION
001370  022706  001134                        CMP     #SWR,R6 ;;DONE?
001374  001374                                BNE     .-6               ;;LOOP BACK IF NO
001376  012706  001100                        MOV     #STACK,SP         ;;SETUP THE STACK POINTER
                                      ;;INITIALIZE A FEW VECTORS
001402  012737  035152  000020                MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001410  012737  000340  000022                MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001416  012737  035412  000030                MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001424  012737  000340  000032                MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001432  012737  040306  000034                MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001440  012737  000340  000036                MOV     #340,@#TRAPVEC+2;LEVEL 7
001446  012737  040364  000024                MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001454  012737  000340  000026                MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001462  013737  033054  033046                MOV     $ENDCT,$EOPCT     ;;SETUP END-OF-PROGRAM COUNTER
001470  005037  035406                        CLR     $TIMES            ;;INITIALIZE NUMBER OF ITERATIONS
001474  005037  035606                        CLR     $ESCAPE           ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001500  112737  000001  001115                MOVB    #1,$ERMAX         ;;ALLOW ONE ERROR PER TEST
001506  012737  001506  001106                MOV     #.,$LPADR         ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001514  012737  001514  001110                MOV     #.,$LPERR         ;;SETUP THE ERROR LOOP ADDRESS
                                      ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                      ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001522  013746  000004                        MOV     @#ERRVEC,-(SP)    ;;SAVE ERROR VECTOR
001526  012737  001562  000004                MOV     #64$,@#ERRVEC     ;;SET UP ERROR VECTOR
001534  012737  177570  001134                MOV     #DSWR,SWR         ;;SETUP FOR A HARDWARE SWICH REGISTER
001542  012737  177570  001136                MOV     #DDISP,DISPLAY    ;;AND A HARDWARE DISPLAY REGISTER
001550  022777  177777  177356                CMP     #-1,@SWR          ;;TRY TO REFERENCE HARDWARE SWR
001556  001012                                BNE     66$               ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
001560  000403                                BR      65$               ;;BRANCH IF NO TIMEOUT
001562  012716  001570                64$:    MOV     #65$,(SP)         ;;SET UP FOR TRAP RETURN
001566  000002                                RTI
001570  012737  000176  001134        65$:    MOV     #SWREG,SWR        ;;POINT TO SOFTWARE SWR
001576  012737  000174  001136                MOV     #DISPREG,DISPLAY
001604  012637  000004                66$:    MOV     (SP)+,@#ERRVEC    ;;RESTORE ERROR VECTOR

001610  005037  001244                        CLR     $PASS             ;;CLEAR PASS COUNT
001614  132737  000200  001257                BITB    #APTSIZE,$ENVM    ;;TEST USER SIZE UNDER APT
001622  001403                                BEQ     67$               ;;YES,USE NON-APT SWITCH
001624  012737  001260  001134                MOV     #$SWREG,SWR       ;;NO,USE APT SWITCH REGISTER
001632                                67$:
001632  104401  040542                        TYPE    ,MSG1             ;TYPE 11/6X DIAGNOSTIC
```

```
                                      ;////////////////////////////////////////////////////////////////
                                              ;THE FOLLOWING ROUTINE IS USED TO INTERROGATE THE PERSON
                                              ;USING THE PROGRAM WHICH NPR DEVICE HE WISHES TO USE
                                      ;////////////////////////////////////////////////////////////////

001636  012700  170000                        MOV     #170000,R0        ;SAVE UBE ADDRESS IF RUNNING UNDER APT
001642  105737  001256                        TSTB    @#$ENV            ;RUNNING UNDER APT?
001646  001410                                BEQ     2$                ;BRANCH IF NO
001650  032777  000400  177256                BIT     #SW08,@SWR        ;ENABLE TESTS USING NPR DEVICES?
001656  001074                                BNE     UBEAPT            ;BRANCH IF YES TO DEFAULT APT DEVICE:UBE
001660  032777  000200  177246                BIT     #SW07,@SWR        ;POWER DOWN TESTS TO BE RUN?
001666  001070                                BNE     UBEAPT            ;BRANCH IF YES TO DEFAULT APT DEVICE:UBE

001670  032777  000400  177236  2$:   BIT     #SW08,@SWR        ;ENABLE TESTS USING NPR DEVICE?
001676  001002                                BNE     Q2                ;BRANCH IF YES

001700  000137  003056                        JMP     START1            ;GO TO BEGINNING OF TESTS

001704  104401  040670                Q2:     TYPE    ,MSG3             ;WHICH DEVICE SHOULD BE USED?
001710  104410                        B2:     RDOCT                     ;WAIT FOR REPLY
001712  012600                                MOV     (SP)+,R0          ;GET ANS OFF STACK
001714  020027  000005                        CMP     R0,#5             ;WAS ANS VALID (<5)?
001720  002002                                BGE     Q1                ;BRANCH IF NO
001722  005700                                TST     R0                ;ANS VALID?
001724  002003                                BGE     B1                ;BRANCH IF YES
001726  104401  041202                Q1:     TYPE    ,MSG4             ;?INVALID ENTRY TRY AGAIN
001732  000766                                BR      B2                ;GO WAIT FOR NEW ANS

001734  000005                        B1:     RESET                     ;INITIALIZE ALL DEVICES
001736  012737  000214  177746                MOV     #214,@#CCR        ;CACHE OFF
001744  006300                                ASL     R0                ;ADJUST FOR WORD INDEXING
001746  000170  001752                        JMP     @TAB(R0)          ;GO ASK FURTHER QUESTIONS ON DEVICE

001752  001764                        TAB:    QUBEN                     ;POINTER TO UNIBUS EXERCISOR (NEW) QUESTIONS
001754  002156                                QUBEO                     ;POINTER TO UNIBUS EXERCISOR (OLD) QUESTIONS
001756  002232                                QRK05                     ;POINTER TO RK05 QUESTIONS
001760  002412                                QRP03                     ;POINTER TO RP03 QUESTIONS
001762  002540                                QTU10                     ;POINTER TO TU10 QUESTIONS

                                      ;////////////////////////////////////////////////////////////////
                                              ;UBE NEW QUESTION AND INITIALIZE ROUTINE
                                      ;////////////////////////////////////////////////////////////////

001764  104401  041242                QUBEN:  TYPE    ,MSG5             ;TYPE THE UBE'S DATA BUFFER ADDRESS
001770  104410                        3$:     RDOCT                     ;WAIT FOR ANS
001772  012737  002006  000004                MOV     #1$,@#4           ;SET UP FOR TIME OUTS
002000  012600                                MOV     (SP)+,R0          ;SEE IF DEVOCE RESPONDS
002002  005710                                TST     (R0)
002004  000413                                BR      2$                ;BRANCH IF YES

002006  012737  000006  000004        1$:     MOV     #6,@#4            ;RESTORE TRAP CATCHER
002014  005037  000006                        CLR     @#6               ;RESTORE TRAP CATCHER
002020  022626                                CMP     (SP)+,(SP)+       ;RESTORE STACK
002022  104401  041312                        TYPE    ,MSG6             ;DEVICE DOES NOT RESPOND; TRAPS TO 4
002026  104401  041202                4$:     TYPE    ,MSG4             ;?INVALID ENTRY, TRY AGAIN
002032  000756                                BR      3$                ;WAIT FOR ANS
```

```
    002034  032700  007417          2$:     BIT     #7417,R0        ;IS ADDRESS LEGAL?
    002040  001372                          BNE     4$              ;BRANCH IF NO
    002042  022700  170000                  CMP     #170000,R0      ;IS ADDRESS LEGAL?
    002046  003367                          BGT     4$              ;BRANCH IF NO
    002050  010001                  UBEAPT: MOV     R0,R1           ;SAVE BUFFER ADDRESS
    002052  042700  177000                  BIC     #177000,R0      ;CALCULATE DEVICE'S
    002056  006200                          ASR     R0              ;INTERRUPT VECTOR
    002060  006200                          ASR     R0
    002062  062700  000510                  ADD     #510,R0         ;R0=DEVICE INT VECTOR
    002066  010037  001226                  MOV     R0,IVEC         ;SAVE DEVICE INT VECTOR
    002072  010137  001224                  MOV     R1,CREG6        ;SAVE DEVICE BUFFER ADDR
    002076  005721                          TST     (R1)+           ;UPDATE ADDRESS
    002100  010137  001222                  MOV     R1,CREG5        ;SAVE UBE CYCLE COUNT REG ADDR.
    002104  005721                          TST     (R1)+           ;UPDATE ADDRESS
    002106  010137  001220                  MOV     R1,CREG4        ;SAVE UBE ADDRESS COUNTER ADDR.
    002112  005721                          TST     (R1)+           ;UPDATE ADDRESS
    002114  010137  001212                  MOV     R1,CREG1        ;SAVE UBE CONTROL REG 1 ADDR.
    002120  005721                          TST     (R1)+           ;UPDATE ADDRESS
    002122  010137  001216                  MOV     R1,CREG3        ;SAVE UBE ERROR CLEAR ADDR.
    002126  005721                          TST     (R1)+           ;UPDATE ADDRESS
    002130  022121                          CMP     (R1)+,(R1)+     ;UPDATE ADDRESS
    002132  010137  001214                  MOV     R1,CREG2        ;SAVE UBE CONTROL REG 2 ADDR.
    002136  013737  001212  001230          MOV     CREG1,EAD       ;SAVE UBE ERROR ADDRESS
    002144  012737  034046  001232          MOV     #HUBEN,SETUP    ;LOAD POINTER FOR UBE HANDLER
    002152  000137  003056                  JMP     START1          ;GO TO BEGINNING OF TEST

                                        ;////////////////////////////////////////////////////////////////////
                                                ;UBE OLD INITIALIZE ROUTINE
                                        ;////////////////////////////////////////////////////////////////////

    002156  012737  002172  000004  QUBEO:  MOV     #1$,@#4         ;SET UP FOR TIME OUTS
    002164  005737  170000                  TST     @#170000        ;SEE IF DATA BUFFER RESPONDS
    002170  000405                          BR      2$
    002172  022626                  1$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
    002174  104401  041312                  TYPE    ,MSG6           ;DEVICE DOESN'T RESPOND
    002200  000137  001726                  JMP     Q1              ;GO CHOOSE ANOTHER DEVICE

    002204  012737  170006  001212  2$:     MOV     #170006,CREG1   ;SAVE THE GO ADDRESS
    002212  012737  034224  001230          MOV     #FAKE,EAD       ;SETUP FAKE ADDRESS FOR ERROR TEST
    002220  012737  034174  001232          MOV     #HUBEO,SETUP    ;LOAD PTER FOR UBE HANDLER
    002226  000137  003056                  JMP     START1          ;GO TO BEGINNING OF TEST

                                        ;////////////////////////////////////////////////////////////////////
                                                ;RK05 QUESTION AND INITIALIZE ROUTINE
                                        ;////////////////////////////////////////////////////////////////////

    002232  012737  002246  000004  QRK05:  MOV     #1$,@#4         ;SET UP FOR TIME OUTS
    002240  005737  177404                  TST     @#RKCS          ;SEE IF RK05 STATUS REG RESPONDS
    002244  000405                          BR      2$

    002246  022626                  1$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
    002250  104401  041312                  TYPE    ,MSG6           ;DEVICE DOES NOT RESPOND
    002254  000137  001726                  JMP     Q1              ;GO CHOOSE ANOTHER DEVICE

    002260  104401  041414          2$:     TYPE    ,MSG7           ;WHICH DRIVE SHOULD BE USED?
```

```
                                                                        ;TYPE 0-7 <CARRIAGE RETURN>
    002264  104410                  4$:     RDOCT                   ;WAIT FOR ANS
    002266  012600                          MOV     (SP)+,R0        ;IS DRIVE VALID # = OR >0?
    002270  002003                          BGE     3$              ;BRANCH IF YES
    002272  104401  041202          5$:     TYPE    ,MSG4           ;INVALID ENTRY, TRY AGAIN
    002276  000772                          BR      4$              ;GO WAIT FOR REPLY

    002300  022700  000010          3$:     CMP     #10,R0          ;IS DRIVE VALID # <7?
    002304  003772                          BLE     5$              ;BRANCH IF NO
    002306  012701  000015                  MOV     #15,R1          ;PUT DRIVE #
    002312  006300                  6$:     ASL     R0              ;IN 3 MSB OF R0
    002314  077102                          SOB     R1,6$           ;LOOP TILL DONE
    002316  010037  001214                  MOV     R0,@#CREG2      ;SAVE DISK ADDRESS REG CONTENTS WITH SELECTED
                                                                        ;DRIVE AND CYLINDER ADDR, SURFACE & SECTOR=0
    002322  012737  177404  001212          MOV     #RKCS,CREG1     ;SAVE THE GO ADDRESS
    002330  012737  177404  001230          MOV     #RKCS,EAD       ;SAVE THE ERROR ADDRESS
    002336  012737  034226  001232          MOV     #HRK05,SETUP    ;LOAD POINTER FOR RK05 HANDLER
    002344  013737  001214  177412          MOV     @#CREG2,@#RKDA  ;SET UP DRIVE #
    002352  012737  000015  177404          MOV     #15,@#RKCS      ;RESET DRIVE
    002360  005001                          CLR     R1              ;INIT COUNT
    002362  032737  000100  177400  8$:     BIT     #100,@#RKDS     ;DRIVE READY?
    002370  001006                          BNE     7$              ;BRANCH IF YES
    002372  005201                          INC     R1              ;WAIT FOR
    002374  001372                          BNE     8$              ;DRIVE RDY
    002376  104401  041645                  TYPE    ,MSG13          ;DEVICE RDY BIT DOES NOT SET
    002402  000137  001726                  JMP     Q1              ;GO CHOOSE ANOTHER DEVICE

    002406  000137  003056          7$:     JMP     START1          ;GO TO FIRST TEST

                                        ;////////////////////////////////////////////////////////////////////
                                                ;RP03 QUESTION AND INITIALIZE ROUTINE
                                        ;////////////////////////////////////////////////////////////////////

    002412  012737  002426  000004  QRP03:  MOV     #1$,@#4         ;SETUP FOR TIME OUT
    002420  005737  176714                  TST     @#RPCS          ;SEE IF RP03 CONTROL REG RESPONDS
    002424  000405                          BR      2$

    002426  022626                  1$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
    002430  104401  041312                  TYPE    ,MSG6           ;DEVICE DOES NOT RESPOND
    002434  000137  001726                  JMP     Q1              ;GO CHOOSE ANOTHER DEVICE

    002440  104401  041414          2$:     TYPE    ,MSG7           ;WHICH DRIVE SHOULD BE USED?
                                                                        ;TYPE 0-7 <CARRIAGE RETURN>
    002444  104410                  4$:     RDOCT                   ;WAIT FOR REPLY
    002446  012600                          MOV     (SP)+,R0        ;GET DRIVE # FROM STACK
    002450  002003                          BGE     3$              ;BRANCH IF DRIVE #>OR=0
    002452  104401  041202          5$:     TYPE    ,MSG4           ;INVALID ENTRY, TRY AGAIN
    002456  000772                          BR      4$              ;GO WAIT FOR REPLY

    002460  022700  000010          3$:     CMP     #10,R0          ;IS DRIVE VALID #> OR=7
    002464  003772                          BLE     5$              ;BRANCH IF NO
    002466  000300                          SWAB    R0              ;PUT DRIVE # IN HIGH BYTE
    002470  010037  001214                  MOV     R0,CREG2        ;SETUP CONTROL MASK WITH DRIVE # AND
    002474  052737  000004  001214          BIS     #4,CREG2        ;A READ OPERATION (NPR DATO)
    002502  005037  176722                  CLR     @#RPCA          ;SETUP CYLINDER ADDRESS REG FOR 0
    002506  005037  176724                  CLR     @#RPDA          ;SETUP DISK ADDRESS REG FOR 0 SECTOR AND TRACK
```

```
002512  012737  176714  001212          MOV     #RPCS,CREG1     ;SAVE THE GO ADDRESS
002520  012737  176714  001230          MOV     #RPCS,EAD       ;SAVE THE ERROR ADDRESS
002526  012737  034460  001232          MOV     #HRP03,SETUP    ;LOAD POINTER TO RP03 HANDLER
002534  000137  003056                  JMP     START1          ;GO TO FIRST TEST

                                ;//////////////////////////////////////////////////////////////////
                                        ;TU10 QUESTION AND INITIALIZE ROUTINE
                                ;//////////////////////////////////////////////////////////////////

002540  012737  002554  000004  QTU10:  MOV     #1$,@#4         ;SETUP FOR TIME OUT
002546  005737  172522                  TST     @#MTC           ;SEE IF TU10 COMMAND REG RESPONDS
002552  000405                          BR      2$              ;YES, BRANCH

002554  022626                  1$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
002556  104401  041312                  TYPE    ,MSG6           ;DEVICE DOES NOT RESPOND
002562  000137  001726                  JMP     Q1              ;GO CHOOSE ANOTHER DEVICE

002566  104401  041414          2$:     TYPE    ,MSG7           ;WHICH DRIVE SHOULD BE USED?
                                                                ;TYPE 0-7 <CARRIAGE RETURN>
002572  104410                  4$:     RDOCT                   ;WAIT FOR REPLY
002574  012600                          MOV     (SP)+,R0        ;GET DRIVE # FROM STACK
002576  002003                          BGE     3$              ;BRANCH IF DRIVE # > OR = 0
002600  104401  041202          5$:     TYPE    ,MSG4           ;INVALID ENTRY TRY AGAIN
002604  000772                          BR      4$              ;WAIT FOR REPLY

002606  022700  000010          3$:     CMP     #10,R0          ;IS DRIVE VALID # < OR = 7
002612  003772                          BLE     5$              ;BRANCH IF NO
002614  000300                          SWAB    R0              ;PUT DRIVE # IN HIGH BYTE
002616  012737  010000  172522          MOV     #10000,@#MTC    ;POWER CLEAR CONTROLLER
002624  012701  000010                  MOV     #10,R1          ;SET DELAY FOR POWER CLEAR
002630  077101                  6$:     SOB     R1,6$           ;WAIT FOR POWER CLEAR
002632  012737  000016  172522          MOV     #16,@#MTC       ;SET UP TO REWIND
002640  050037  172522                  BIS     R0,@#MTC        ;SET UP DRIVE # IN CONTROL
002644  012701  000777                  MOV     #777,R1         ;SET UP DELAY COUNT
002650  077101                  7$:     SOB     R1,7$           ;DELAY FOR SELECT REMOTE
002652  032737  000100  172520          BIT     #100,@#MTS      ;SEE IF DRIVE SELECTED
002660  001003                          BNE     8$              ;BRANCH IF YES
002662  104401  041507                  TYPE    ,MSG10          ;DRIVE NOT SELECTED PROPERLY
002666  000744                          BR      5$              ;SELECT ANOTHER

002670  032737  000004  172520  8$:     BIT     #4,@#MTS        ;WRITE PROTECT ON?
002676  001403                          BEQ     9$              ;BRANCH IF NO
002700  104401  041546                  TYPE    ,MSG11          ;WRITE PROTECT ON
002704  000735                          BR      5$              ;SELECT ANOTHER UNIT

002706  005237  172522          9$:     INC     @#MTC           ;REWIND TAPE
002712  032737  000001  172520  10$:    BIT     #1,@#MTS        ;TAPE UNIT RDY?
002720  001774                          BEQ     10$             ;LOOP TILL IS
002722  012737  034714  001232          MOV     #HTU10,SETUP    ;LOAD PTER TO TU10 HANDLER
002730  012737  172522  001212          MOV     #MTC,CREG1      ;SAVE GO ADDRESS
002736  012737  172522  001230          MOV     #MTC,EAD        ;SAVE ERROR ADDRESS
002744  012737  040000  001214          MOV     #40000,CREG2    ;SET UP CONTROL MASK WITH DENSITY=800BPI, 7 CHANNEL
002752  050037  001214                  BIS     R0,CREG2        ;SET DRIVE # IN MASK

                                        ;NOW WRITE MIN # OF BYTES ON TAPE (24)8

```

```
002756  013737  001214  172522          MOV     CREG2,@#MTC     ;SET UP TO DO WRITE
002764  052737  000004  172522          BIS     #4,@#MTC        ;SET FUNCTION=WRITE
002772  012737  177760  172524          MOV     #-20,@#MTBRC    ;WRITE (20)8 BYTES
003000  012737  060000  172526          MOV     #BUFL,@#MTCMA   ;SETUP ADDRESS FOR XFER
003006  005237  172522                  INC     @#MTC           ;START WRITE
003012  012701  177777                  MOV     #177777,R1      ;SET UP FOR MAX DELAY
003016  032737  000001  172520  12$:    BIT     #1,@#MTS        ;UNIT DONE?
003024  001005                          BNE     11$             ;BRANCH IF YES
003026  077105                          SOB     R1,12$          ;LOOP TILL MAX COUNT DONE
003030  104401  041645                  TYPE    ,MSG13          ;DEVICE RDY BIT DOES NOT SET
003034  000137  001704                  JMP     Q2              ;TRY ANOTHER DEVICE

003040  005737  172522          11$:    TST     @#MTC           ;ERROR BIT SET?
003044  100004                          BPL     START1          ;BRANCH IF NO TO FIRST TEST
003046  104401  041614                  TYPE    ,MSG12          ;DEVICE ERROR BIT SET
003052  000137  001704                  JMP     Q2              ;TRY ANOTHER DEVICE


003056  012737  033352  000004  START1: MOV     #UT4,@#4        ;SETUP FOR UNEXPECTED TRAPS TO VECTOR 4
003064  012737  033142  000114          MOV     #UPERR,@#114    ;SET UP FOR UNEXPECTED PARITY ERRORS.
003072  042737  000001  177572          BIC     #1,@#MMR0       ;KT OFF IF ON
003100  012706  001100                  MOV     #STACK,SP       ;INIT STACK POINTER

003104  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
003106  076600                          MED                     ;GET CONTENTS OF LOG REG
003110  000022                          .WORD   RLOG
003112  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
003116  076600                          MED                     ;UNLOCK ERROR LOG
003120  000222                          .WORD   WLOG
003122  012600                          MOV     (SP)+,R0        ;RESTORE R0

003124  023727  001232  034046          CMP     SETUP,#HUBEN    ;IS THERE A UNIBUS EXERCISER DEVICE?
003132  001013                          BNE     1$              ;BRANCH IF NO
003134  013737  001226  001172          MOV     IVEC,$TMP0      ;GET ITS VECTOR
003142  062737  000002  001172          ADD     #2,$TMP0        ;AND PUT A TRAP
003150  013777  001172  176050          MOV     $TMP0,@IVEC     ;CATCHER THERE
003156  005077  176010                  CLR     @$TMP0
003162                          1$:

                                ;;*************************************************************
                                ;*TEST 1        TEST PA MUX AND PHYSICAL ADDRESS DRIVERS
                                ;*
                                ;*IF THE INHIBIT TESTS USING KT SWITCH (SW12)=1, THIS
                                ;*TEST IS INHIBITED.
                                ;*    THE PHYSICLAL ADDRESS LINES A17,A16,A15 ARE CHECKED
                                ;*THAT THEY CAN CHANGE STATES.  THE MEMORY IS FIRST SIZED
                                ;*TO SEE IF THERE IS MORE THAN 16K OF MEMORY.  IF NO, THIS
                                ;*TEST IS SKIPPED.  IF THERE IS MORE THAN 16K OF
                                ;*MEMORY, THE HIGH ADDRESS BITS A17, A16, A15 WILL BE TESTED
                                ;*WITH A FLOAT 1, 0 PATTERN.
                                ;*    WHEN AN ADDRESS IS FOUND TO CONTAIN INCORRECT DATA
                                ;*AN ERROR MESSAGE IS TYPED.  IN ADDITION, A HANDLER (NSSYN)
                                ;*FOR TRAPS TO VECTOR 4 WILL REPORT OTHER ADDRESSING ERRORS.

                                ;;*************************************************************
003162  012737  000214  177746  TST1:   MOV     #214,@#CCR      ;TURN OFF CACHE FOR SCOPE
```

```
003170  000004                                 $COPE
003172  012737  004164  001234                 MOV     #TST2,$KTST       ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
003200  032777  010000  175726                 BIT     #SW12,@SWR        ;INHIBIT TESTS USING KT?
003206  001402                                 BEQ     A15               ;BRANCH IF NO
003210  000137  004164                         JMP     TST2              ;YES,GO TO NEXT TEST
003214  012737  004072  000004  A15:           MOV     #NSSYN,@#4        ;SET UP FOR TRAPS TO 4 DUE TO ADDRESSING ERRORS

                                                       ;START CHECK OF NON PROGRAM LOCATIONS

003222  052737  000200  036034                 BIS     #200,@#KY11       ;TURN ON KT FOR $SIZE
003230  004737  035750                         JSR     PC,$SIZE          ;SIZE MEMORY
003234  022737  001000  036322                 CMP     #1000,$LSTBK      ;IS THERE MORE THAN 16K OF MEM?
003242  003402                                 BLE     A16               ;BRANCH IF YES
003244  000137  004054                         JMP     A17               ;NO,GO  EXIT TEST
003250  012700  100000          A16:           MOV     #100000,R0        ;SET UP R0 TO ADDRESS PAR4
003254  012701  077000                         MOV     #77000,R1         ;INITIALIZE TEST DATA REG
003260  012737  077406  172310                 MOV     #77406,@#KIPDR4   ;PAGE LENGTH=4K, EXPAND UP READ/WRITE
003266  012737  001000  172350                 MOV     #1000,@#KIPAR4    ;SET UP TO TEST ADDRESS BIT 15
003274  005237  177572                         INC     @#MMR0            ;TURN ON KT
003300  023737  036322  172350  A5:            CMP     $LSTBK,@#KIPAR4   ;TESTED ALL ADDRESSES?
003306  001401                                 BEQ     A3                ;BRANCH IF AT LAST ONE
003310  101411                                 BLOS    A4                ;BRANCH IF PAST LAST ADDRESS

                                               ;SAVE CONTENTS OF ADDRESSES TESTING ON STACK AND PUT TEST DATA IN THEM

003312  011046                  A3:            MOV     (R0),-(SP)        ;SAVE DATA
003314  010110                                 MOV     R1,(R0)           ;WRITE TEST DATA IN LOC
003316  005201                                 INC     R1                ;CALC NEW TEST DATA
003320  006337  172350                         ASL     @#KIPAR4          ;CALC NEXT TEST ADDRESS
003324  005737  172350                         TST     @#KIPAR4          ;AT LAST ADDRESS?
003330  001401                                 BEQ     A4                ;GO TEST DATA IF PAST LAST ADDR.
003332  000762                                 BR      A5                ;GO SEE IF ADDR. TO BE TESTED

                                               ;SEE IF DATA AT ADDRESSES

003334  012701  077000          A4:            MOV     #77000,R1         ;INIT. TEST DATA REG
003340  012737  001000  172350                 MOV     #1000,@#KIPAR4    ;INIT PAR FOR LOWEST ADDR.
003346  023737  036322  172350  A0:            CMP     $LSTBK,@#KIPAR4   ;LOOKED AT LAST ADDRESS?
003354  001401                                 BEQ     A6                ;BRANCH IF AT LAST
003356  101474                                 BLOS    A77               ;BRANCH IF PAST ADDRESS
003360  020110                  A6:            CMP     R1,(R0)           ;WAS DATA IN LOC?
003362  001007                                 BNE     A1                ;BRANCH IF NO TO ERROR
003364  005201                                 INC     R1                ;CALC. TEST DATA
003366  006337  172350                         ASL     @#KIPAR4          ;CALC. NEXT TEST LOC.
003372  005737  172350                         TST     @#KIPAR4          ;AT LAST ADDR.?
003376  001464                                 BEQ     A77               ;BRANCH IF DONE WITH HIGH ADDR.
003400  000762                                 BR      A0                ;LOOK AT NEXT LOCATION

003402  011037  001164          A1:            MOV     (R0),$REG3        ;SAVE BAD DATA

                                               ;ROUTINE TO CONVERT VIRTUAL ADDRESS IN R0 TO PHYSICAL ADDRESS IN R4,R5

003406  010002                                 MOV     R0,R2             ;GET VIRTUAL ADDRESS
003410  005003                                 CLR     R3                ;INIT SHIFT COUNTER
003412  006202                  1$:            ASR     R2                ;SHIFT BLOCK NO. TO LSB 0-6
003414  005203                                 INC     R3                ;COUNT SHIFTS
```

```
003416  020327  000006                         CMP     R3,#6             ;ALL DONE?
003422  001373                                 BNE     1$                ;BRANCH IF NO
003424  010204                                 MOV     R2,R4             ;SAVE BLOCK #
003426  042704  177600                         BIC     #177600,R4        ;CALC. BLOCK #
003432  006202                  2$:            ASR     R2                ;SHIFT ACTIVE PAGE FIELD TO LSB 1-3
003434  005203                                 INC     R3                ;COUNT SHIFTS
003436  020327  000014                         CMP     R3,#14            ;ALL DONE?
003442  001373                                 BNE     2$                ;BRANCH IF NO
003444  042702  177761                         BIC     #177761,R2        ;CALC. APFX2
003450  062702  172340                         ADD     #KIPAR0,R2        ;CALC. ADDR. OF PAR REFERENCING
003454  011202                                 MOV     (R2),R2           ;GET (PAR)
003456  060204                                 ADD     R2,R4             ;CALC. PHYSICAL BLOCK #
003460  010405                                 MOV     R4,R5             ;START TO SAVE PHYSICAL ADDR. A17,A16
003462  005003                                 CLR     R3                ;INIT. SHIFT COUNTER
003464  006205                  3$:            ASR     R5                ;SHIFT ADDR BIT 17,16 TO LSB 0,1
003466  005203                                 INC     R3                ;COUNT
003470  020327  000012                         CMP     R3,#12            ;DONE?
003474  001373                                 BNE     3$                ;BRANCH IF NO
003476  005003                                 CLR     R3                ;INIT SHIFT COUNTER
003500  006304                  4$:            ASL     R4                ;SHIFT MSB TO BIT 16
003502  005203                                 INC     R3                ;COUNT
003504  020327  000006                         CMP     R3,#6             ;ALL DONE?
003510  001373                                 BNE     4$                ;BRANCH IF NO
003512  010002                                 MOV     R0,R2             ;GET VIRTUAL ADDRESS
003514  042702  177700                         BIC     #177700,R2        ;LEAVE BLOCK COUNT IN REG
003520  060204                                 ADD     R2,R4             ;HAVE R4 CONTAIN PHY. ADDR. 0-15
003522  010437  001162                         MOV     R4,$REG2          ;SAVE LO ADD
003526  010537  001160                         MOV     R5,$REG1          ;SAVE HI ADD
003532  010137  001166                         MOV     R1,$REG4          ;SAVE CURRENT DATA
003536  012706  001100                         MOV     # STACK,SP        ;RESTORE STACK IF LOOP
003542  104020                                 ERROR   20                ;ERROR: PHYSICAL ADDRESS LINE ERROR
                                                                         ;       ADDRESS HELD WRONG DATA
003544  000137  004054                         JMP     A17               ;GO TO NEXT TEST

                                               ;RESTORE FLOATING "1" ADDRESSES

003550  012737  004000  172350  A77:           MOV     #4000,@#KIPAR4    ;INIT. KIPAR1 TO RESTORE 3 LOC
003556  012700  100000                         MOV     #100000,R0        ;INIT R0 TO ADDRESS KIPAR4
003562  022737  004000  036322                 CMP     #4000,@#$LSTBK    ;WERE 3 LOC WRITTEN?
003570  101405                                 BLOS    A80               ;BRANCH IF YES
003572  022737  002000  036322                 CMP     #2000,@#$LSTBK    ;WERE 2 LOC WRITTEN?
003600  101402                                 BLOS    A81               ;BRANCH IF YES
003602  000405                                 BR      A82               ;RESTORE LAST LOC ONLY
003604  012610                  A80:           MOV     (SP)+,(R0)        ;
003606  012737  002000  172350  A81:           MOV     #2000,@#KIPAR4    ;SET UP KIPAR4 TO RESTORE 2 LOC
003614  012610                                 MOV     (SP)+,(R0)
003616  012737  001000  172350  A82:           MOV     #1000,@#KIPAR4    ;SET UP KIPAR4 TO RESTORE LAST LOC
003624  012610                                 MOV     (SP)+,(R0)

                                               ;NOW TEST ADDRESS 15,16,17 CAN FLOAT A '0'

003626  022737  003740  036322                 CMP     #3740,$LSTBK      ;ENOUGH MEM TO TEST A17?
003634  003107                                 BGT     A17               ;BRANCH IF NO
003636  012701  177000                         MOV     #177000,R1        ;SET UP TEST DATA
003642  012700  103776                         MOV     #103776,R0        ;ADDR. PAR4 & HAVE ALL LOW ADDRESS BITS=1
003646  012737  003740  172350                 MOV     #3740,@#KIPAR4    ;SET UP PAR4 SO A17=0 A16,A15=1 & ALL HIGH ADDR. BITS =1
```

```
003654  011046                          MOV     (R0),-(SP)       ;SAVE DATA ON STACK
003656  010110                          MOV     R1,(R0)          ;LOAD TEST ADDRESS WITH DATA
003660  005201                          INC     R1               ;CHANGE DATA
003662  022737  005740  036322          CMP     # 5740,$LSTBK    ;ENOUGH MEM TO TEST A16?
003670  003006                          BGT     A10              ;BRANCH IF NO
003672  012737  005740  172350          MOV     # 5740,@#KIPAR4  ;HAVE A17,A16,A15=101
003700  011046                          MOV     (R0),-(SP)       ;SAVE DATA
003702  010110                          MOV     R1,(R0)          ;LOAD TEST DATA
003704  005201                          INC     R1               ;CHANGE DATA

003706  022737  006740  036322  A10:    CMP     # 6740,$LSTBK    ;ENOUGH MEM TO TEST A15?
003714  003005                          BGT     A12              ;BRANCH IF NO
003716  012737  006740  172350          MOV     # 6740,@#KIPAR4
003724  011046                          MOV     (R0),-(SP)       ;SAVE DATA
003726  010110                          MOV     R1,(R0)          ;LOAD TEST DATA

                                        ;SEE IF DATA WRITTEN PROPERLY

003730  012737  003740  172350  A12:    MOV     # 3740,@#KIPAR4 ;SET UP ADDRESS
003736  012701  177000                  MOV     #177000,R1
003742  020110                          CMP     R1,(R0)          ;DATA OK?
003744  001402                          BEQ     A11              ;BRANCH IF YES
003746  000137  003402                  JMP     A1               ;REPORT ERROR

003752  022737  005740  036322  A11:    CMP     # 5740,$LSTBK    ;TESTING A16?
003760  003034                          BGT     A14              ;BRANCH IF NO TO RESTORE DATA
003762  005201                          INC     R1               ;UPDATE DATA
003764  012737  005740  172350          MOV     # 5740,@#KIPAR4 ;SETUP ADDRESS
003772  020110                          CMP     R1,(R0)          ;DATA OK?
003774  001402                          BEQ     A13              ;BRANCH YES
003776  000137  003402                  JMP     A1               ;REPORT ERROR

004002  022737  006740  036322  A13:    CMP     # 6740,$LSTBK    ;TESTING A15?
004010  003014                          BGT     A85              ;BRANCH NO TO RESTORE DATA
004012  005201                          INC     R1               ;UPDATE DATA
004014  012737  006740  172350          MOV     # 6740,@#KIPAR4 ;SETUP ADDRESS
004022  020110                          CMP     R1,(R0)          ;DATA OK?
004024  001402                          BEQ     A86              ;BRANCH YES
004026  000137  003102                  JMP     A1               ;REPORT ERROR

                                        ;RESTORE DATA

004032  012610                  A86:    MOV     (SP)+,(R0)       ;RESTORE 3 LOCS
004034  012737  005740  172350          MOV     # 5740,@#KIPAR4
004042  012610                  A85:    MOV     (SP)+,(R0)       ;RESTORE 2 LOCS
004044  012737  003740  172350          MOV     # 3740,@#KIPAR4
004052  012610                  A14:    MOV     (SP)+,(R0)       ;RESTORE 1 LOC

                                        ;EXIT TEST

004054  042737  000001  177572  A17:    BIC     #1,@#MMR0        ;TURN OFF KT IF ON
004062  012737  033352  000004          MOV     #UT4,@#4         ;RESTORE HANDLER FOR UNEXPECTED TRAPS
004070  000435                          BR      TST2             ;;GO TO NEXT TEST

                                ;ROUTINE TO HANDLE NO SSYN ERRORS

```

```
004072  010046                  NSSYN:  MOV     R0,-(SP)
004074  076600                  1$:     MED
004076  000101                          .WORD   HIADD
004100  010037  001160                  MOV     R0,@#$REG1
004104  076600                          MED
004106  000102                          .WORD   LOADD
004110  010037  001162                  MOV     R0,@#$REG2
004114  012600                          MOV     (SP)+,R0
004116  022626                          CMP     (SP)+,(SP)+
004120  013737  177744  001164          MOV     @#EREG,@#$REG3
004126  104021                          ERROR   21               ;ERROR: TRAP TO VECTOR 4  WHEN TESTING PHYSICAL ADDR. LI
004130  012737  033352  000004          MOV     #UT4,@#4         ;RESTORE HANDLER FOR UNEXPECT. TRAPS
004136  042737  000001  177572          BIC     #1,@#MMR0        ;TURN OFF KT

004144  010046                          MOV     R0,-(SP)         ;SAVE R0 FOR MED INST
004146  076600                          MED                      ;GET CONTENTS OF LOG REG
004150  000022                          .WORD   RLOG
004152  052700  100001                  BIS     #100001,R0       ;ENABLE ERROR LOG & LOG FIRST MODE
004156  076600                          MED                      ;UNLOCK ERROR LOG
004160  000222                          .WORD   WLOG
004162  012600                          MOV     (SP)+,R0         ;RESTORE R0



                                ;;*****************************************************************
                                ;*TEST 2         TEST CACHE CAN BE TURNED OFF AND HIT REG CLEARED
                                ;*
                                ;*   THE CACHE IS TURNED OFF AND THE CACHE CONTROL REG
                                ;*IS CHECKED TO CONTAIN ALL 1'S FOR ALL SETTABLE BITS
                                ;*EXCEPT BIT6 (WWP),NEXT THE HIT REG (HMR) IS TESTED TO BE ALL 0'S.  AFTER THIS,
                                ;*A LOW CACHE ADDRESS AND THEN A HIGH ADDRESS ARE TRIED TO
                                ;*BE MADE HITS AND THEN THE HMR IS CHECKED TO BE ALL 0'S,
                                ;*(LOW CACHE ADDRESS HAS PHYSICAL ADDRESS BIT 10=0).
                                ;*
                                ;*IF THIS TEST REPORTS A FATAL ERROR, ALL FOLLOWING TESTS
                                ;*ARE ABORTED

                                ;;*****************************************************************
004164  012737  000214  177746  TST2:   MOV     #214,@#CCR       ;CACHE OFF FOR SCOPE
004172  000004                          SCOPE
004174  012737  000214  177746          MOV     #214,@#CCR               ;SET UP DATA
004202  012737  004320  001234          MOV     #TST3,$KTST      ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
004210  013737  177746  001160          MOV     @#CCR,$REG1      ;GET (CCR)
004216  022737  000214  001160          CMP     #214,@#$REG1     ;WERE BITS SET IN CCR?
004224  001406                          BEQ     T01L01           ;BRANCH IF YES
004226  012737  000214  001162          MOV     #214,$REG2       ;SAVE GOOD DATA
004234  104005                          ERROR   5                ;FATAL ERROR: CACHE CONTROL REG HELD WRONG DATA
004236  000137  033020                  JMP     $EOP             ;ABORT TEST

004242  013737  177752  001160  T01L01: MOV     @#HMR,$REG1      ;SEE IF HIT MISS REG HAS ALL MISSES
004250  001405                          BEQ     T01L02           ;BRANCH IF YES
004252  005037  001162          T01L03: CLR     $REG2            ;SAVE GOOD DATA
004256  104006                          ERROR   6                ;FATAL ERROR:HIT/MISS REG HELD WRONG DATA
004260  000137  033020                  JMP     $EOP             ;ABORT TEST

004264  012700  060000          T01L02: MOV     #BUFL,R0         ;INITIALIZE R0 TO LOW ADDRESS
```

```
004270  021010                           CMP     (R0),(R0)          ;TRY TO MAKE LOC A HIT
004272  013737  177752  001160           MOV     @#HMR,$REG1        ;SEE IF MISS ON LOW ADDRESS SPACE
004300  001364                           BNE     T01L03             ;BRANCH IF GOT FALSE HIT
004302  012700  062000                   MOV     #BUFH,R0           ;SET R0=TO HIGH ADDRESS SPACE
004306  021010                           CMP     (R0),(R0)          ;TRY TO MAKE HIGH ADDRESS A HIT
004310  013737  177752  001160           MOV     @#HMR,$REG1        ;SEE IF MISS AT HIGH ADDRESS
004316  001355                           BNE     T01L03             ;BRANCH IF GET FALSE HIT

                                 ;;*************************************************************
                                 ;*TEST 3         TEST CAN GET A HIT ON A HIGH CACHE ADDRESS AND HIT REG CAN =1
                                 ;*
                                 ;*   THIS IS THE FIRST TEST WHERE THE HIGH HALF OF CACHE IS
                                 ;*TURNED ON.  THE CACHE CONTROL REG IS FIRST LOADED AND CHECKED
                                 ;*TO CONTAIN THE PROPER VALUE.  THEN ONE LOCATION IN CACHE
                                 ;*IS MADE A HIT.  THE HIT REG IS THEN TESTED TO MAKE SURE
                                 ;*ITS 5 MSB CAN =1 AT THE CORRECT TIME.
                                 ;*
                                 ;*IF THIS TEST REPORTS A FATAL ERROR, ALL FOLLOWING TESTS
                                 ;*ABORTED.

                                 ;;*************************************************************
004320  012737  000214  177746   TST3:   MOV     #214,@#CCR         ;CACHE OFF FOR SCOPE
004326  000004                           SCOPE
004330  012737  004616  001234           MOV     #TST4,$KTST        ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
004336  012737  000204  177746           MOV     #204,@#CCR         ;TURN ON HIGH ADDRESSES OF CACHE
004344  013700  177746                   MOV     @#CCR,R0           ;GET (CCR)
004350  022700  000204                   CMP     #204,R0            ;WAS CACHE TURNED ON?
004354  001413                           BEQ     T02L01             ;BRANCH IF YES
004356  042737  000014  177746           BIC     #14,@#CCR          ;TURN CACHE OFF
004364  010037  001160                   MOV     R0,$REG1           ;SAVE BAD DATA
004370  012737  000210  001162           MOV     #210,$REG2         ;SAVE GOOD DATA
004376  104005                   1$:     ERROR   5                  ;FATAL ERROR: CACHE CONTROL REG HELD WRONG DATA
004400  000137  033020                   JMP     $EOP               ;ABORT TEST

004404  012701  177752           T02L01: MOV     #HMR,R1            ;SAVE HIT/MISS ADDRESS
004410  012700  062000                   MOV     #BUFH,R0           ;INITIALIZE R0 TO HIGH ADDRESS
004414  021010                           CMP     (R0),(R0)          ;MAKE ADDRESS A HIT
004416  011102                           MOV     (R1),R2            ;SAVE HIT-MISS REG SHIFTED ONE
004420  011103                           MOV     (R1),R3            ;SAVE HIT MISS REG SHIFTED TWO
004422  011104                           MOV     (R1),R4            ;SAVE HIT MISS REG SHIFTED THREE
004424  011105                           MOV     (R1),R5            ;SAVE HIT MISS REG SHIFTED FOUR
004426  052737  000014  177746           BIS     #14,@#CCR          ;TURN OFF CACHE
004434  030227  000002                   BIT     R2,#HMR1           ;DID WE GET A HIT AND WAS IT SHIFTED?
004440  001010                           BNE     T02L02             ;BRANCH IF YES
004442  010237  001160                   MOV     R2,$REG1           ;SAVE BAD DATA
004446  012737  000002  001162           MOV     #2,$REG2           ;SAVE GOOD DATA
004454  104013                   T02L06: ERROR   13                 ;FATAL ERROR:HIT/MISS REG HELD WRONG DATA
004456  000137  033020                   JMP     $EOP               ;ABORT TEST

004462  030327  000004           T02L02: BIT     R3,#HMR2           ;WAS DATA SHIFTED?
004466  001006                           BNE     T02L03             ;BRANCH IF YES
004470  010337  001160                   MOV     R3,$REG1           ;SAVE BAD DATA
004474  012737  000004  001162           MOV      #4,$REG2          ;SAVE GOOD DATA
004502  000764                           BR      T02L06             ;REPORT ERROR

004504  030427  000010           T02L03: BIT     R4,#HMR3           ;WAS DATA SHIFTED?
```

```
004510  001006                           BNE     T02L04             ;BRANCH IF YES
004512  010437  001160                   MOV     R4,$REG1           ;SAVE BAD DATA
004516  012737  000010  001162           MOV     #10,$REG2          ;SAVE GOOD DATA
004524  000753                           BR      T02L06             ;REPORT ERROR

004526  030527  000020           T02L04: BIT     R5,#HMR4           ;WAS DATA SHIFTED?
004532  001006                           BNE     T02L05             ;BRANCH IF YES
004534  010537  001160                   MOV     R5,$REG1           ;SAVE BAD DATA
004540  012737  000020  001162           MOV     #20,$REG2          ;SAVE GOOD DATA
004546  000742                           BR      T02L06             ;REPORT ERROR

004550  012737  000204  177746   T02L05: MOV     #204,@#CCR         ;TURN HALF CACHE ON
004556  021010                           CMP     (R0),(R0)          ;MAKE ADDRESS A HIT
004560  021010                           CMP     (R0),(R0)          ;SHIFT HIT 3 TIMES
004562  000240                           NOP                        ;SHIFT HIT FOURTH TIME
004564  011102                           MOV     (R1),R2            ;SHIFT HIT FIFTH TIME AND SAVE

004566  030227  000040                   BIT     R2,#HMR5           ;WAS DATA SHIFTED?
004572  001011                           BNE     TST4               ;;BRANCH IF YES TO NEXT TEST
004574  052737  000014  177746           BIS     #14,@#CCR          ;TURN CACHE OFF
004602  010237  001160                   MOV     R2,$REG1           ;SAVE BAD DATA
004606  012737  000054  001162           MOV     #54,$REG2          ;SAVE GOOD DATA
004614  000717                           BR      T02L06             ;REPORT ERROR

                                 ;;*************************************************************
                                 ;*TEST 4         TEST FORCE MISS ON HIGH ADDRESS
                                 ;*
                                 ;*A LOCATION IS PUT IN CACHE, CACHE IS THEN TURNED OFF
                                 ;*AND THE LOCATION IS CHECKED TO BE A MISS.

                                 ;;*************************************************************
004616  012737  000214  177746   TST4:   MOV     #214,@#CCR         ;TURN OFF CACHE FOR SCOPE
004624  000004                           SCOPE
004626  012737  004712  001234           MOV     #TST5,$KTST        ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
004634  012737  000204  177746           MOV     #204,@#CCR         ;TURN ON HIGH ADDRESS OF CACHE
004642  012700  062000                   MOV     #BUFH,R0           ;INITIALIZE R0=HIGH ADDRESS
004646  021010                           CMP     (R0),(R0)          ;MAKE LOC A HIT
004650  052737  000014  177746           BIS     #14,@#CCR          ;TURN OFF CACHE
004656  005710                           TST     (R0)               ;SEE IF LOC STILL A HIT
004660  033727  177752  000004           BIT     @#HMR,#HMR2        ;WAS IT A MISS?
004666  001411                           BEQ     TST5               ;;BRANCH IF YES
004670  013737  177746  001160           MOV     @#CCR,$REG1        ;SAVE (CCR)
004676  012737  000000  001162           MOV     #0,$REG2           ;SAVE PHYSICAL ADDRESS HIGH
004704  010037  001164                   MOV     R0,$REG3           ;SAVE PHYSICAL ADDRESS LOW
004710  104012                   1$:     ERROR   12                 ;ERROR:FORCE MISS BIT FAILED TO CAUSE MISS.


                                 ;;*************************************************************
                                 ;*TEST 5         TEST CACHE TRACKS WHEN CACHE IS OFF
                                 ;*
                                 ;*   A LOC IS MADE A HIT IN CACHE, CACHE IS THEN TURNED OFF
                                 ;*AND A SECOND LOC IS REFERENCED WHICH HAS AN OVERLAPPING
                                 ;*CACHE ADDRESS WITH THE FIRST ONE.  CACHE IS TURNED ON
                                 ;*AND THE SECOND LOC IS TESTED TO BE A HIT (IMPLYING
                                 ;*CACHE HAS TRACKED).

                                 ;;*************************************************************
```

```
 004712  012737  000214  177746  TST5:   MOV     #214,@#CCR       ;CACHE OFF FOR SCOPE
 004720  000004                          SCOPE
 004722  012737  005044  001234          MOV     #TST6,$KTST      ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 004730  012737  000204  177746          MOV     #204,@#CCR       ;HALF CACHE ON
 004736  023737  002000  002000          CMP     @#2000,@#2000            ;PUT DATA IN CACHE
 004744  033727  177752  000004          BIT     @#HMR,#HMR2      ;DATA IN CACHE?
 004752  001423                          BEQ     1$               ;BRANCH IF NO TO ERROR
 004754  052737  000014  177746          BIS     #14,@#CCR        ;CACHE OFF
 004762  005737  062000                  TST     @#BUFH           ;REFERENCE LOC NOT IN CACHE AND SEE IF TRACK
 004766  012737  000204  177746          MOV     #204,@#CCR       ;HALF CACHE ON
 004774  005737  062000                  TST     @#BUFH           ;SEE IF CACHE TRACKED
 005000  033727  177752  000004          BIT     @#HMR,#HMR2      ;HIT?
 005006  001016                          BNE     TST6             ;;YES, GO TO NEXT TEST

 005010  052737  000014  177746          BIS     #14,@#CCR        ;CACHE OFF
 005016  104107                          ERROR   107              ;ERROR: CACHE DID NOT TRACK WHEN FORCE MISS ON
 005020  000411                          BR      TST6             ;;GO TO NEXT TEST

 005022  052737  000014  177746  1$:     BIS     #14,@#CCR        ;CACHE OFF
 005030  005037  001160                  CLR     $REG1            ;SAVE BAD ADDR.
 005034  012737  002000  001162          MOV     #2000,$REG2      ;SAVE BAD ADDR.
 005042  104043                          ERROR   43               ;ERROR: ADDRESS COULD NOT BE MADE A HIT

                                 ;;*************************************************************
                                 ;*TEST 6          TEST DATOB OPERATION
                                 ;*
                                 ;*    A DATOB IS DONE TO AN ADDRESS NOT IN CACHE AND THEN
                                 ;*THE LOC IS REFERENCED TO SEE THAT CACHE WAS NOT ALLOCATED.
                                 ;*NEXT A DATOB IS DONE TO AN ODD LOC IN CACHE AND THE
                                 ;*CORRECT BYTE IS CHECKED TO BE MODIFIED.  THIS IS RE-
                                 ;*PEATED FOR AN EVEN ADDRESS.

                                 ;;*************************************************************
 005044  012737  000214  177746  TST6:   MOV     #214,@#CCR       ;TURN OFF CACHE FOR SCOPE
 005052  000004                          SCOPE
 005054  012737  005364  001234          MOV     #TST7,$KTST      ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 005062  012737  000204  177746          MOV     #204,@#CCR       ;TURN ON CACHE HIGH ADDRESS
 005070  005737  002000                  TST     @#2000           ;MAKE LOC BUFH IN NEXT INST. A MISS
 005074  112737  000377  062000          MOVB    #377,@#BUFH      ;DO DATOB TO NON-HIT LOC TO SEE IT DOESN'T GET CACHED
 005102  005737  062000                  TST     @#BUFH           ;SEE IF DATA PUT IN CACHE
 005106  033727  177752  000004          BIT     @#HMR,#HMR2      ;WAS DATA A HIT?
 005114  001413                          BEQ     T04L01           ;BRANCH IF NO
 005116  052737  000014  177746          BIS     #14,@#CCR        ;TURN OFF CACHE
 005124  012737  000000  001160          MOV     #0,$REG1         ;SAVE PHYSICAL ADDRESS HIGH
 005132  012737  062000  001162          MOV     #BUFH,$REG2      ;SAVE NO HIT PHYSICAL ADDRESS LOW
 005140  104007                  1$:     ERROR   7                ;ERROR:DATA CACHED ON DATOB TO NO 'HIT' ADD.
 005142  000510                          BR      TST7             ;;GO TO NEXT TEST

 005144  005037  062000          T04L01: CLR     @#BUFH           ;INITIALIZE LOC BUFH
 005150  112737  177777  062001          MOVB    #177777,@#BUFH+1 ;DO DATOB TO A HIT LOC
 005156  005737  062000                  TST     @#BUFH           ;SEE IF DATA PUT IN CACHE
 005162  033727  177752  000004          BIT     @#HMR,#HMR2      ;WAS DATA A HIT ?
 005170  001013                          BNE     T04L02           ;BRANCH IF YES
 005172  052737  000014  177746          BIS     #14,@#CCR        ;TURN OFF CACHE
 005200  012737  000000  001160          MOV     #0,$REG1         ;SAVE PHYSICAL ADDRESS HIGH
 005206  012737  062000  001162          MOV     #BUFH,$REG2      ;SAVE PHYSICAL ADDRESS LOW
```

```
 005214  104010                  1$:     ERROR   10               ;ERROR:DATA NOT CACHED ON DATOB TO A 'HIT' LOC.
 005216  000462                          BR      TST7             ;;GO TO NEXT TEST

 005220  022737  177400  062000  T04L02: CMP     #177400,@#BUFH   ;WAS DATA WRITTEN CORRECTLY?
 005226  001424                          BEQ     T04L03           ;BRANCH IF YES
 005230  013700  062000                  MOV     @#BUFH,R0        ;GET BAD DATA
 005234  052737  000014  177746          BIS     #14,@#CCR        ;TURN OFF CACHE
 005242  012737  000000  001160          MOV     #0,$REG1         ;SAVE PHYSICAL ADDRESS HIGH
 005250  012737  062000  001162          MOV     #BUFH,$REG2      ;SAVE PHYSICAL ADDRESS LOW
 005256  010037  001164                  MOV     R0,$REG3         ;SAVE BAD DATA
 005262  012737  177400  001166          MOV     #177400,$REG4    ;SAVE GOOD DATA
 005270  104011                  1$:     ERROR   11               ;ERROR:CACHE DID NOT CONTAIN PROPER DATA ON DATOB
 005272  042737  000010  177746          BIC     #10,@#CCR        ;TURN CACHE ON

 005300  005037  062000          T04L03: CLR     @#BUFH           ;INITIALIZE LOCATION
 005304  112737  000377  062000          MOVB    #377,@#BUFH      ;DO DATOB TO EVEN ADDRESS
 005312  022737  000377  062000          CMP     #377,@#BUFH      ;WAS DATA WRITTEN CORRECTLY?
 005320  001421                          BEQ     TST7             ;;BRANCH IF YES TO NEXT TEST
 005322  013700  062000                  MOV     @#BUFH,R0        ;GET BAD DATA
 005326  052737  000014  177746          BIS     #14,@#CCR        ;TURN CACHE OFF
 005334  012737  000000  001160          MOV     #0,$REG1         ;SAVE PHYSICAL ADDRESS HIGH
 005342  012737  062000  001162          MOV     #BUFH,$REG2      ;SAVE PHYSICAL ADDRESS LOW
 005350  010037  001164                  MOV     R0,$REG3         ;SAVE BAD DATA
 005354  012737  000377  001166          MOV     #377,$REG4       ;SAVE GOOD DATA
 005362  104011                  1$:     ERROR   11               ;ERROR:CACHE DID NOT CONTAIN PROPER DATA ON DATOB.

                                 ;;*************************************************************
                                 ;*TEST 7          TEST DATO ALLOCATES CACHE
                                 ;*
                                 ;*    A LOC IS MADE A HIT IN CACHE, THEN A DATO IS DONE TO
                                 ;*A SECOND CACHE ADDRESS WITH ADDRESS BITS A0-A10 THE SAME.
                                 ;*THE SECOND ADDRESS IS THEN CHECKED TO BE ALLOCATED IN
                                 ;*CACHE.

                                 ;;*************************************************************
 005364  012737  000214  177746  TST7:   MOV     #214,@#CCR       ;CACHE OFF FOR SCOPE
 005372  000004                          SCOPE
 005374  012737  006000  001234          MOV     #TST10,$KTST     ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 005402  012737  000204  177746          MOV     #204,@#CCR       ;HALF CACHE ON
 005410  023737  002000  002000          CMP     @#2000,@#2000    ;PUT LOC IN CACHE TO MAKE NEXT REF A MISS
 005416  033727  177752  000004          BIT     @#HMR,#HMR2      ;HIT?
 005424  001422                          BEQ     T05L01           ;BRANCH TO ERROR IF NO
 005426  005037  062000                  CLR     @#BUFH           ;DO DATO TO A MISS ADDRESS
 005432  005737  062000                  TST     @#BUFH           ;LOC IN CACHE?
 005436  033727  177752  000004          BIT     @#HMR,#HMR2      ;HIT?
 005444  001023                          BNE     T05L02           ;;YES, GO TO END OF TEST
 005446  052737  000014  177746          BIS     #14,@#CCR        ;CACHE OFF
 005454  005037  001160                  CLR     $REG1            ;SAVE FAILING ADDRESS
 005460  012737  062000  001162          MOV     #BUFH,$REG2      ;SAVE FAILING ADDRESS
 005466  104014                          ERROR   14               ;ERROR:  ADDR. NOT A HIT AFTER DATO TO IT
 005470  000411                          BR      T05L02           ;;GO TO END OF TEST

 005472  052737  000014  177746  T05L01: BIS     #14,@#CCR        ;CACHE OFF
 005500  005037  001160                  CLR     $REG1            ;SAVE FAILING ADDR
 005504  012737  002000  001162          MOV     #2000,$REG2      ;SAVE FAILING ADDR
 005512  104043                          ERROR   43               ;ERROR: ADDR. COULD NOT BE MADE A HIT
 005514  052737  000014  177746  T05L02: BIS     #14,@#CCR       ;CACHE OFF WHEN CROSS CACHE ADDRESS BOUNDARY
 005522  000526                          BR      TST10           ;;GO TO NEXT TEST


         006000                          .=6000                  ;ADJUST ADDRESS SPACE FOR NEXT TEST


                                         ;;*************************************************************
                                         ;*TEST 10       TEST CAN GET HIT AND  FORCE MISS ON LOW CACHE ADDRESS
                                         ;*
                                         ;*   THIS IS THE FIRST TEST WHERE LOW CACHE IS TURNED
                                         ;*ON.  THE CACHE CONTROL REG IS FIRST LOADED AND CHECKED
                                         ;*TO CONTAIN THE PROPER VALUE.  THEN ONE LOC IN LOW
                                         ;*CACHE IS MADE A HIT.  THE HIT IS CHECKED FOR AND THEN
                                         ;*CACHE IS TURNED OFF AND THE LOC IS RETESTED TO NOW BE
                                         ;*A MISS.
                                         ;*
                                         ;*IF THIS TEST REPORTS A FATAL ERROR, ALL FOLLOWING TESTS
                                         ;*ARE ABORTED.

                                         ;;*************************************************************
 006000  012737  000214  177746  TST10:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
 006006  000004                          SCOPE
 006010  012737  006166  001234          MOV     #TST11,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 006016  012737  000210  177746          MOV     #210,@#CCR      ;TURN ON LOW CACHE
 006024  013700  177746                  MOV     @#CCR,R0        ;GET (CCR)
 006030  022700  000210                  CMP     #210,R0         ;(CCR) OK?
 006034  001413                          BEQ     2$              ;BRANCH IF YES
 006036  052737  000014  177746          BIS     #14,@#CCR       ;CACHE OFF
 006044  010037  001160                  MOV     R0,$REG1        ;SAVE BAD DATA
 006050  012737  000210  001162          MOV     #210,$REG2      ;SAVE GOOD DATA
 006056  104005                          ERROR   5               ;FATAL ERROR: CCR HELD WRONG DATA
 006060  000137  033020                  JMP     $EOP            ;ABORT PROGRAM

 006064  012700  060000          2$:     MOV     #BUFL,R0        ;INIT R0=LOW ADDRESS
 006070  021010                          CMP     (R0),(R0)       ;MAKE LOC A HIT
 006072  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS IT A HIT?
 006100  001012                          BNE     4$              ;BRANCH IF YES
 006102  052737  000014  177746          BIS     #14,@#CCR       ;CACHE OFF
 006110  005037  001160                  CLR     $REG1           ;SAVE ADDRESS
 006114  012737  060000  001162          MOV     #BUFL,$REG2     ;SAVE ADDRESS
 006122  104043                          ERROR   43              ;ERROR: ADDRESS COULD NOT BE MADE A HIT
 006124  000420                          BR      TST11           ;;GO TO NEXT TEST

 006126  052737  000014  177746  4$:     BIS     #14,@#CCR       ;CACHE OFF
 006134  005710                          TST     (R0)            ;SEE IF LOC STILL A HIT
 006136  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS IT A MISS?
 006144  001410                          BEQ     TST11           ;;BRANCH IF YES
 006146  013737  177746  001160          MOV     @#CCR,$REG1     ;SAVE (CCR)
 006154  005037  001162                  CLR     $REG2           ;SAVE ADDRESS
 006160  010037  001164                  MOV     R0,$REG3        ;SAVE ADDRESS
 006164  104012                          ERROR   12              ;ERROR: FORCE MISS BIT FAILED TO CAUSE MISS


                                         ;;*************************************************************
                                         ;*TEST 11       TEST OF TAG ADDRESS COMPARATOR
```

```
                                         ;*
                                         ;*   THIS TEST USES ONE LOC IN CACHE AND LOADS IT WITH
                                         ;*VARIOUS TAG ADDRESSES.  A GROUP OF MEMORY REFERENCES
                                         ;*ARE MADE FOR EACH TAG ADDRESS AND IT IS DETERMINED
                                         ;*WHETHER EACH REFERENCE WILL BE A HIT OR A MISS.  THE
                                         ;*LOW ADDRESS COMPARATOR FOR BITS A11-A14 IS TESTED FIRST.
                                         ;*A TAG ADDRESS IS LOADED AND THEN ALL POSSIBLE COMBINATIONS
                                         ;*OF MEMORY ADDRESSES TO THAT CHIP ARE MADE.  ALL TAG
                                         ;*COMBINATIONS ARE TRIED IN THIS MANNER FOR THESE LOW
                                         ;*ADDRESSES.  THE HIGH ADDRESS COMPARATOR FOR BITS A15-
                                         ;*A17 IS HELD CONSTANT DURING THIS TIME.  THE SAME PRO-
                                         ;*CEDURE IS REPEATED FOR THIS HIGH ADDRESS COMPARATOR
                                         ;*WHILE THE LOW ONE IS HELD CONSTANT.  THE COMPARATOR
                                         ;*TEST IS LIMITED TO THE MEMORY SIZE AVAILABLE.
                                         ;*   KIPAR4 CONTAINS THE TAG ADDRESS BEING TESTED.  KIPAR5
                                         ;*CONTAINS THE MEMORY REFERENCE ADDRESS BEING MADE.  IF
                                         ;*INHIBIT TESTS USING KT SWITCH IS SET (SW12), THIS TEST
                                         ;*IS INHIBITED.

                                         ;;*************************************************************
 006166  012737  000214  177746  TST11:  MOV     #214,@#CCR              ;TURN OFF CACHE FOR SCOPE
 006174  000004                          SCOPE
 006176  012737  006626  001234          MOV     #TST12,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 006204  032777  010000  172722          BIT     #SW12,@SWR      ;INHIBIT TESTS USING KT?
 006212  001402                          BEQ     2$              ;BRANCH IF NO
 006214  000137  006626                  JMP     @#TST12         ;GO TO NEXT TEST
 006220  052737  000200  036034  2$:     BIS     #200,@#$KT11    ;TURN ON KT FOR MEM SIZING
 006226  004737  035750                  JSR     PC,$SIZE        ;SIZE MEM
 006232  012700  172350                  MOV     #KIPAR4,R0      ;SET UP TO
 006236  012701  172310                  MOV     #KIPDR4,R1      ;INIT KIPDR4, 5 & KIPAR4, 5
 006242  005020                  1$:     CLR     (R0)+           ;FOR TESTING
 006244  012721  077406                  MOV     #77406,(R1)+    ;PAGE LENGTH=4K, EXPAND UP, READ/WRITE
 006250  020127  172314                  CMP     R1,#KIPDR6      ;KT SET UP?
 006254  001372                          BNE     1$              ;BRANCH IF NO
 006256  000403                          BR      T06L12          ;GO TO START OF TEST

 006260  005737  172352          T06L01: TST     @#KIPAR5        ;PAST MAX PAR5?
 006264  001404                          BEQ     T06L03          ;BRANCH IF YES TO CHOOSE NEXT TAG ADDRESS
 006266  023737  172352  036322  T06L12: CMP     @#KIPAR5,@#$LSTBK  ;REFERENCED ALL POSSIBLE ADDRS. FOR THIS COMPARATOR?
 006274  003434                          BLE     T06L02          ;BRANCH IF NO
 006276  023727  172350  001000  T06L03: CMP     @#KIPAR4,#1000  ;TESTED COMPARATOR FOR ADDRESS BITS 15,16,17?
 006304  002404                          BLT     T06L05          ;BRANCH IF NO
 006306  062737  001000  172350          ADD     #1000,@#KIPAR4  ;TEST NEXT ADDRESS BIT OF HIGH ADDR. COMP.
 006314  000403                          BR      T06L06

 006316  062737  000040  172350  T06L05: ADD     #40,@#KIPAR4    ;TEST NEXT ADDRESS BIT OF LOW ADDR. COMP.
 006324  005737  172350          T06L06: TST     @#KIPAR4        ;PAST MAX TAG ADDRESS?
 006330  001533                          BEQ     T06L04          ;GO TO END OF TEST IF YES
 006332  023737  172350  036322          CMP     @#KIPAR4,@#$LSTBK  ;HAVE ALL POSSIBLE TAG INPUTS TO COMPARATOR BEEN DONE
 006340  002127                          BGE     T06L04          ;GO TO END OF TEST IF YES
 006342  023727  172350  001000          CMP     @#KIPAR4,#1000  ;ARE WE TESTING THE HIGH ADDRESS COMPARATOR?
 006350  002003                          BGE     T06L07          ;BRANCH IF YES
 006352  005037  172352                  CLR     @#KIPAR5        ;INIT PAR5 TO TEST LOW ADDR. COMP.
 006356  000403                          BR      T06L02          ;GO TEST COMPARATOR

 006360  012737  001000  172352  T06L07: MOV     #1000,@#KIPAR5  ;INIT. PAR5 TO TEST HIGH ADDR. COMP.
```

```
006360  052737  000014  177746  T06L02: BIS     #14,@#CCR       ;TURN CACHE OFF
006374  012737  120000  001172          MOV     #120000,$TMP0   ;START CALC. OF PHYSICAL
006402  004737  033434                  JSR     PC,VIP          ;ADDRESS REFERENCING AND
006406  013700  172350                  MOV     @#KIPAR4,R0     ;START CALC OF TAG ADDRESS TESTING
006412  005001                          CLR     R1              ;GET TAG FIELD TO 7 LSB R0
006414  006200                  1$:     ASR     R0              ;GET TAG FIELD TO 7 LSB R0
006416  005201                          INC     R1              ;GET TAG FIELD TO 7 LSB R0
006420  020127  000005                  CMP     R1,#5           ;GET TAG FIELD TO 7 LSB R0
006424  001373                          BNE     1$              ;GET TAG FIELD TO 7 LSB R0
006426  010037  001164                  MOV     R0,$REG3        ;SAVE TAG IN CASE OF ERROR

006432  052737  000001  177572          BIS     #1,@#MMR0       ;TURN ON KT
006440  012737  000210  177746  T06L08: MOV     #210,@#CCR      ;TURN ON HALF OF CACHE ON
006446  023737  172350  172352          CMP     @#KIPAR4,@#KIPAR5       ;WILL REFERENCE BE A HIT
006454  001422                          BEQ     T06L09          ;BRANCH IF YES
006456  023737  100000  120000          CMP     @#100000,@#120000  ;LOAD ADDRESS IN TAG FIELD & THEN REFERENCE IT
006464  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS REFERENCE A MISS?
006472  001435                          BEQ     T06L10          ;BRANCH IF YES
006474  052737  000014  177746          BIS     #14,@#CCR       ;TURN OFF CACHE
006502  012737  006440  001110          MOV     #T06L08,@#$LPERR  ;INIT. FOR LOOP ON ERROR
006510  104022                          ERROR   22              ;ERROR: TEST OF ADDR. COMP. FAILED TO BE MISS
006512  042737  000001  177572          BIC     #1,@#MMR0       ;TURN OFF KT
006520  000442                          BR      TST12           ;;GO TO NEXT TEST

006522  023737  100000  120000  T06L09: CMP     @#100000,@#120000  ;LOAD ADDRESS IN TAG FIELD & THEN REFERENCE IT
006530  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS REF. A HIT?
006536  001013                          BNE     T06L10          ;BRANCH IF YES
006540  052737  000014  177746          BIS     #14,@#CCR       ;TURN OFF CACHE FOR ERROR REPORT
006546  012737  006440  001110          MOV     #T06L08,@#$LPERR  ;SETUP RETURN FOR LOOP ON ERROR
006554  104023                          ERROR   23              ;ERROR: TEST OF ADDR. COMP. FAILED TO BE HIT
006556  042737  000001  177572          BIC     #1,@#MMR0       ;TURN OFF KT
006564  000420                          BR      TST12           ;;GO TO NEXT TEST

006566  023727  172352  000740  T06L10: CMP     @#KIPAR5,#740   ;REFERENCED ADDRESSES OF LOWER ADDR. COMP.?
006574  001640                          BEQ     T06L03          ;BRANCH IF YES
006576  002404                          BLT     T06L11          ;BRANCH IF PAR5 STILL REF. LOW ADDR. COMP.
006600  062737  001000  172352          ADD     #1000,@#KIPAR5  ;ADDRESS NEXT LOC FOR HIGH ADDR. COMPARATOR
006606  000624                          BR      T06L01          ;SEE IF DONE
006610  062737  000040  172352  T06L11: ADD     #40,@#KIPAR5    ;ADDRESS NEXT LOC FOR LOW ADDR. COMP.
006616  000620                          BR      T06L01          ;SEE IF DONE

006620  042737  000001  177572  T06L04: BIC     #1,@#MMR0       ;TURN KT OFF

                                        ;;*************************************************************
                                        ;*TEST 12       TEST FORCE MISS LOCKS OUT PARITY ERRORS & CCR WWP CAN =1
                                        ;*
                                        ;*   THIS IS THE FIRST TEST WHERE WRITE WRONG PARITY AND
                                        ;*THE CACHE PARITY TRAP IS EXERCISED.  FIRST THE WWP IS
                                        ;*SET AND THE CACHE CONTROL REG IS CHECKED TO CONTAIN THE
                                        ;*PROPER VALUE.  A PARITY TRAP IS THEN FORCED AND TESTED
                                        ;*FOR.  THE LOCATION IS REWRITTEN WITH WRONG PARITY AND
                                        ;*THEN THE CACHE IS TURNED OFF.  THE LOCATION IS REFERENCED
                                        ;*AND NO PARITY TRAP WHEN FORCE MISS IS ON IS CHECKED FOR.

                                        ;;*************************************************************
006626  012737  000214  177746  TST12:  MOV     #214,@#CCR      ;TURN OFF CACHE
```

```
006634  000004                          SCOPE
006636  012737  007140  001234          MOV     #TST13,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
006644  012737  006770  000114          MOV     #T09L01,@#PVEC  ;SETUP PARITY TRAP HANDLER
006652  012737  000310  177746          MOV     #310,@#CCR      ;TURN ON HALF OF CACHE & WWP
006660  013700  177746                  MOV     @#CCR,R0
006664  020027  000310                  CMP     R0,#310         ;WERE BITS SET IN CCR?
006670  001414                          BEQ     T09L02          ;BRANCH IF YES
006672  012737  000014  177746          MOV     #14,@#CCR       ;TURN CACHE OFF
006700  010037  001160                  MOV     R0,$REG1        ;SAVE BAD DATA
006704  012737  000310  001162          MOV     #310,$REG2      ;SAVE GOOD DATA
006712  104026                          ERROR   26              ;ERROR: CACHE CONTROL REG HELD WRONG DATA
006714  012737  000310  177746          MOV     #310,@#CCR      ;TURN ON HALF OF CACHE & WWP

006722  005037  060000          T09L02: CLR     @#BUFL          ;WRITE WRONG PARITY IN 1 LOC
006726  012737  000210  177746          MOV     #210,@#CCR      ;WWP OFF
006734  005737  060000                  TST     @#BUFL          ;SEE IF GET PARITY TRAP


                                        ;RID CACHE OF BAD PARITY
006740  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
006746  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE


006752  005037  001160                  CLR     $REG1           ;SAVE ADDRESS
006756  012737  060000  001162          MOV     #BUFL,$REG2     ;SAVE ADDRESS
006764  104042                          ERROR   42              ;ERROR: NO PARITY TRAP FROM LOC WRITTEN WITH WRONG PARIT
006766  000450                          BR      T09L06          ;GO TO END OF TEST

006770                          T09L01:

                                        ;RID CACHE OF BAD PARITY
006770  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
006776  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE



007002  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
007004  076600                          MED                     ;GET CONTENTS OF LOG REG
007006  000022                          .WORD   RLOG
007010  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
007014  076600                          MED                     ;UNLOCK ERROR LOG
007016  000222                          .WORD   WLOG
007020  012600                          MOV     (SP)+,R0        ;RESTORE R0

007022  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
007024  012737  007072  000114          MOV     #T09L03,@#PVEC  ;SET UP PARITY TRAP HANDLER
007032  012737  000310  177746          MOV     #310,@#CCR      ;TURN HALF OF CACHE ON & WWP
007040  005037  060000                  CLR     @#BUFL          ;WRITE WRONG PARITY IN ONE LOC
007044  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF
007052  005737  060000                  TST     @#BUFL          ;SEE IF SEE GET PARITY TRAP

007056                          T09L04:

                                        ;RID CACHE OF BAD PARITY
007056  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
007064  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE
```

```


007070  000407                          BR      T09L06          ;GO TO END OF TEST

007072                          T09L03:

                                        ;RID CACHE OF BAD PARITY
007072  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
007100  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE


007104  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
007106  104024                          ERROR   24              ;ERROR: FORCE MISS DID NOT INHIBIT PARITY ERRORS

007110                          T09L06:

007110  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
007112  076600                          MED                     ;GET CONTENTS OF LOG REG
007114  000022                          .WORD   RLOG
007116  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
007122  076600                          MED                     ;UNLOCK ERROR LOG
007124  000222                          .WORD   WLOG
007126  012600                          MOV     (SP)+,R0        ;RESTORE R0

007130  012737  033142  000114          MOV     #UPERR,@#PVEC   ;RESTORE HANDLER FOR UNEXPECTED PARITY ERRORS
007136  000400                          BR      TST13           ;;GO TO NEXT TEST


                                ;;*****************************************************************
                                ;*TEST 13        TEST OF TAG PARITY GENERATOR/CHECKER
                                ;*
                                ;*   THIS TEST INITIALLY SIZES MEMORY TO DETERMINE THE
                                ;*MAXINUM TESTABLE ADDRESS.  KIPAR4 IS SETUP TO WRITE ALL
                                ;*TAG COMBINATIONS UP TO THE MAX ADDRESS INTO ONE CACHE
                                ;*LOCATION.  FIRST, THE LOCATION IS WRITTEN WITH WRONG
                                ;*PARITY FOR ALL THE TAG COMBINATIONS AND A PARITY TRAP
                                ;*IS FORCED AND TESTED FOR.  AFTER EACH TRAP, THE PROGRAM
                                ;*CHECKS THAT THE TRAP WAS FROM THE TAG FIELD AND THAT
                                ;*THE TAG CONTENTS (FROM ERROR LOG) WAS WHAT WAS WRITTEN.
                                ;*THIS LATTER CHECK IS DONE PRIMARILY TO ENSURE THAT THE
                                ;*TRAP WAS BECAUSE WRONG PARITY WAS WRITTEN AND NOT DUE
                                ;*TO A FAILING LOCATION.
                                ;*   SECOND, THE LOCATION IS WRITTEN WITH GOOD PARITY FOR
                                ;*ALL TAG COMBINATIONS.  THE LOC IS REFERENCED AND ANY
                                ;*PARITY ERROR IS DETECTED AND REPORTED.
                                ;*   IF INHIBIT TESTS USING KT SWITCH (SW12) IS SET,
                                ;*THIS TEST IS INHIBITED.

                                ;;*****************************************************************
007140  012737  000214  177746  TST13:  MOV     #214,@#CCR      ;TURN CACHE OFF FOR SCOPE
007146  000004                          SCOPE
007150  012737  010230  001234          MOV     #TST14,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
007156  032777  010000  171750          BIT     #SW12,@SWR      ;INHIBIT TEST USING KT11?
007164  001402                          BEQ     1$              ;BRANCH IF NO
007166  000137  010230                  JMP     @#TST14         ;GO TO NEXT TEST
007172  052737  000200  036034  1$:     BIS     #200,@#$KT11    ;TURN ON KT FOR $SIZE
```

```
007200  004737  035750                  JSR     PC,$SIZE        ;SIZE MEMORY
007204  012737  007354  000114          MOV     #T07L01,@#PVEC  ;SET UP TO HANDLE PARITY TRAPS
007212  012737  077406  172310          MOV     #77406,@#KIPDR4 ;PAGE LENGTH=4K, EXPAND UP, READ/WRITE
007220  005037  172350                  CLR     @#KIPAR4        ;INIT PAR
007224  052737  000001  177572          BIS     #1,@#MMR0       ;TURN KT ON
007232  023737  172350  036322  T07L04: CMP     @#KIPAR4,@#$LSTBK  ;TESTED ALL POSSIBLE ADDRESSES?
007240  003402                          BLE     1$              ;BRANCH IF NO TO CONTINUE
007242  000137  007650                  JMP     T07L02          ;TEST GOOD PARITY GEN.
007246  012737  000310  177746  1$:     MOV     #310,@#CCR      ;TURN HALF OF CACHE ON & WWP

007254  013737  100000  100000  T07L03: MOV     @#100000,@#100000  ;WRITE WRONG PARITY IN LOC
007262  012737  000210  177746          MOV     #210,@#CCR      ;WWP OFF
007270  005737  100000                  TST     @#100000        ;FORCE A PARITY ERROR


                                        ;RID CACHE OF BAD PARITY
007274  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
007302  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE


007306  012737  100000  001172          MOV     #100000,$TMP0   ;GET ADDRESS JUST TESTED
007314  004737  033434                  JSR     PC,VIP          ;CALC ITS PHYSICAL ADDRESS
007320  013737  172350  001172          MOV     @#KIPAR4,$TMP0  ;GET PAR FOR TAG CALC.
007326  004737  033606                  JSR     PC,TAG          ;CALC WHAT TAG CONTENTS SHOULD BE
007332  013737  001172  001164          MOV     $TMP0,$REG3     ;SAVE (TAG) SHOULD BE
007340  012737  007232  001110          MOV     #T07L04,@#$LPERR        ;SET UP RETURN FOR LOOP ON ERROR
007346  104027                          ERROR   27              ;ERROR: TEST OF TAG PARITY GENERATOR/CHECKER FAILED
                                                                ;       DID NOT GET PARITY TRAP FROM TAG FIELD
                                                                ;        WHEN WROTE WRONG PARITY
007350  000137  010214                  JMP     @#T07L05        ;GO TO END OF TEST

007354                          T07L01:

                                        ;RID CACHE OF BAD PARITY
007354  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
007362  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE



007366  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
007370  076600                          MED                     ;GET CONTENTS OF LOG REG
007372  000022                          .WORD   RLOG
007374  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
007400  076600                          MED                     ;UNLOCK ERROR LOG
007402  000222                          .WORD   WLOG
007404  012600                          MOV     (SP)+,R0        ;RESTORE R0

007406  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
007410  032737  000040  177744          BIT     #40,@#EREG      ;TRAP DUE TO PARITY ERROR IN TAG?
007416  001040                          BNE     T07L06          ;BRANCH IF YES
007420  076600                          MED                     ;GET LOG INFORMATION
007422  000102                          .WORD   LOADD
007424  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
007430  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
007432  000101                          .WORD   RSER
007434  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
```

```
 007436  042700  177776                   BIC     #177776,R0        ;ONLY LOOK AT A17, A16
 007442  010037  001160                   MOV     R0,$REG1          ;SAVE ADDRESS
 007446  076600                           MED                       ;GET TAG LOG INFO.
 007450  000107                           .WORD   RTAG
 007452  000300                           SWAB    R0                ;PUT TAG IN LOW BYTE
 007454  042700  177400                   BIC     #177400,R0        ;LOOK AT TAG ONLY
 007460  010037  001164                   MOV     R0,$REG3          ;SAVE TAG
 007464  013737  172350  001172           MOV     @#KIPAR4,$TMP0    ;GET PAR FOR TAG CALC.
 007472  004737  033606                   JSR     PC,TAG            ;FIND GOOD CONTENTS OF TAG
 007476  013737  001172  001166           MOV     $TMP0,$REG4       ;SAVE GOOD DATA
 007504  012737  007232  001110           MOV     #T07L04,@#$LPERR  ;SET UP RETURN FOR ERROR LOOP
 007512  104030                           ERROR   30                ;ERROR: TEST OF TAG PARITY GEN/CHECKER FAILED
                                                                    ;       DID NOT GET PARITY TRAP FROM TAG FIELD
                                                                    ;       WHEN WROTE WRONG PARITY
 007514  000137  010214                   JMP     @#T07L05          ;GO TO END OF TEST

 007520  013737  172350  001172  T07L06: MOV     @#KIPAR4,$TMP0    ;GET PAR FOR TAG CALC.
 007526  004737  033606                   JSR     PC,TAG            ;CALC WHAT TAG SHOULD BE
 007532  076600                           MED                       ;GET TAG LOG INFO.
 007534  000107                           .WORD   RTAG
 007536  000300                           SWAB    R0                ;PUT TAG IN LOW BYTE
 007540  042700  177400                   BIC     #177400,R0        ;LOOK AT TAG ONLY
 007544  020037  001172                   CMP     R0,$TMP0          ;DATA OK?
 007550  001432                           BEQ     T07L07            ;BRANCH IF YES
 007552  010037  001164                   MOV     R0,$REG3          ;SAVE TAG
 007556  076600                           MED                       ;GET LOG INFORMATION
 007560  000102                           .WORD   LOADD
 007562  010037  001162                   MOV     R0,$REG2          ;SAVE INFORMATION
 007566  076600                           MED                       ;GET LOG INFOR FOR PHY. ADDR. A17,A16
 007570  000101                           .WORD   RSER
 007572  000300                           SWAB    R0                ;PUT PHY. ADDR A17, A16 IN LOW BYTE
 007574  042700  177776                   BIC     #177776,R0        ;ONLY LOOK AT A17, A16
 007600  010037  001160                   MOV     R0,$REG1          ;SAVE ADDRESS
 007604  013737  001172  001166           MOV     $TMP0,@#$REG4     ;SAVE GOOD DATA
 007612  012737  007232  001110           MOV     #T07L04,@#$LPERR  ;SET UP RETURN FOR ERROR LOOP
 007620  104031                           ERROR   31                ;ERROR: TEST OF TAG PARITY GEN/CHECKER FAILED
                                                                    ;       TAG FIELD HELD WRONG DATA ON PARITY TRAP
 007622  123727  001103  000003           CMPB    @#$ERFLG,#3       ;MORE THAN THREE ERRORS?
 007630  101402                           BLOS    T07L07            ;BRANCH IF NO
 007632  000137  010214                   JMP     T07L05            ;GO TO END OF TEST

 007636  062737  000040  172350  T07L07: ADD     #40,@#KIPAR4      ;CALC NEXT TAG ADDRESS TO TEST
 007644  000137  007232                   JMP     T07L04            ;CONTINUE TEST

 007650                          T07L02:

                                          ;RID CACHE OF BAD PARITY
 007650  012737  000214  177746           MOV     #214,@#CCR        ;CACHE OFF IF ON
 007656  004737  035134                   JSR     PC,SWEEP          ;GO PURGE CACHE


 007662  012737  007734  000114           MOV     #T07L08,@#PVEC    ;SET UP FOR PARITY ERRORS
 007670  005037  172350                   CLR     @#KIPAR4          ;INIT ADDRESSES
 007674  023737  172350  036322  T07L09: CMP     @#KIPAR4,@#$LSTBK ;TESTED ALL POSSIBLE ADDRESSES?
 007702  003144                           BGT     T07L05            ;YES GO TO END OF TEST
 007704  012737  000210  177746           MOV     #210,@#CCR        ;TURN HALF CACHE ON
```

```
 007712  013737  100000  100000           MOV     @#100000,@#100000 ;GENERATE PARITY IN CACHE
 007720  005737  102000                   TST     @#102000          ;CHECK PARITY IN CACHE
 007724  062737  000040  172350           ADD     #40,@#KIPAR4      ;CALC NEXT TAG ADDRESS TO TEST
 007732  000760                           BR      T07L09            ;CONTINUE TEST

 007734                          T07L08:

                                          ;RID CACHE OF BAD PARITY
 007734  012737  000214  177746           MOV     #214,@#CCR        ;CACHE OFF IF ON
 007742  004737  035134                   JSR     PC,SWEEP          ;GO PURGE CACHE



 007746  010046                           MOV     R0,-(SP)          ;SAVE R0 FOR MED INST
 007750  076600                           MED                       ;GET CONTENTS OF LOG REG
 007752  000022                           .WORD   RLOG

 007754  052700  100001                   BIS     #100001,R0        ;ENABLE ERROR LOG & LOG FIRST MODE
 007760  076600                           MED                       ;UNLOCK ERROR LOG
 007762  000222                           .WORD   WLOG
 007764  012600                           MOV     (SP)+,R0          ;RESTORE R0

 007766  022626                           CMP     (SP)+,(SP)+       ;RESTORE STACK
 007770  076600                           MED                       ;GET LOG INFOR FOR PHY. ADDR. A17,A16
 007772  000101                           .WORD   RSER
 007774  000300                           SWAB    R0                ;PUT PHY. ADDR A17, A16 IN LOW BYTE
 007776  042700  177776                   BIC     #177776,R0        ;ONLY LOOK AT A17, A16
 010002  010037  001160                   MOV     R0,$REG1          ;SAVE ADDRESS
 010006  076600                           MED                       ;GET LOG INFORMATION
 010010  000102                           .WORD   LOADD
 010012  010037  001162                   MOV     R0,$REG2          ;SAVE INFORMATION
 010016  032737  000040  177744           BIT     #40,@#EREG        ;ERROR DUE TO TAG ERROR?
 010024  001424                           BEQ     T07L10            ;BRANCH IF NO
 010026  076600                           MED                       ;GET TAG LOG INFO.
 010030  000107                           .WORD   RTAG
 010032  000300                           SWAB    R0                ;PUT TAG IN LOW BYTE
 010034  042700  177400                   BIC     #177400,R0        ;LOOK AT TAG ONLY
 010040  010037  001164                   MOV     R0,$REG3          ;SAVE TAG
 010044  013737  172350  001172           MOV     @#KIPAR4,$TMP0    ;GET PAR FOR TAG CALC.
 010052  004737  033606                   JSR     PC,TAG            ;CALC GOOD DATA
 010056  013737  001172  001166           MOV     $TMP0,@#$REG4     ;SAVE GOOD DATA
 010064  012737  007674  001110           MOV     #T07L09,@#$LPERR  ;SET UP FOR ERROR LOOP
 010072  104034                           ERROR   34                ;ERROR: TEST OF TAG PARITY GEN/CHECKER FAILED
                                                                    ;       PARITY ERROR OCCURRED IN TAG FIELD
 010074  000447                           BR      T07L05            ;GO TO END OF TEST

 010076  032737  000100  177744  T07L10: BIT     #100,@#EREG       ;ERROR IN LOW BYTE?
 010104  001414                           BEQ     T07L11            ;BRANCH IF NO
 010106  076600                           MED                       ;GET LOG INFORMATION
 010110  000106                           .WORD   CDL
 010112  010037  001164                   MOV     R0,$REG3          ;SAVE INFORMATION
 010116  013737  102000  001166           MOV     @#102000,@#$REG4  ;SAVE GOOD DATA
 010124  012737  007674  001110           MOV     #T07L09,@#$LPERR  ;INIT LOOP ON ERROR RETURN
 010132  104033                           ERROR   33                ;ERROR: TEST OF TAG PARITY GEN/CHECKER FAILED
                                                                    ;       PARITY ERROR IN LOW BYTE OF DATA
 010134  000427                           BR      T07L05            ;GO TO END OF TEST

```

```
010136  032737  000200  177744  T07L11: BIT     #200,@#EREG     ;ERROR IN HIGH BYTE?
010144  001414                          BEQ     T07L12          ;BRANCH IF NO
010146  076600                          MED                     ;GET LOG INFORMATION
010150  000106                          .WORD   CDH
010152  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
010156  013737  102000  001166          MOV     @#102000,$REG4  ;SAVE GOOD DATA
010164  012737  007674  001110          MOV     #T07L09,@#$LPERR  ;SET UP LOOP ON ERROR
010172  104032                          ERROR   32              ;ERROR: TEST OF TAG PARITY GEN/CHECKER FAILED
                                                                ;       PARITY ERROR IN HIGH BYTE OF DATA
010174  000407                          BR      T07L05          ;GO TO END OF TEST

010176  016637  177774  001164  T07L12: MOV     -4(SP),$REG3    ;SAVE PC OF ERROR
010204  012737  007674  001110          MOV     #T07L09,@#$LPERR  ;SET UP FOR ERROR LOOP
010212  104001                          ERROR   1               ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE

010214  042737  000001  177572  T07L05: BIC     #1,@#MNR0       ;TURN KT OFF
010222  012737  033142  000114          MOV     #UPERR,@#114    ;RESTORE UNEXPECTED PARITY ERROR HANDLER

                                        ;;*****************************************************************
                                        ;*TEST 14        TEST OF DATA PARITY GENERATOR/CHECKER
                                        ;*
                                        ;*   WRONG PARITY IS WRITTEN INTO ONE BYTE OF ONE LOCATION
                                        ;*IN THE CACHE DATA FIELD VIA A DATOB,  THE LOC IS REFERENCED
                                        ;*AND THE PARITY TRAP IS CHECKED FOR,  THE TRAP FROM THE
                                        ;*CORRECT BYTE IS THEN TESTED,  THIS PROCEDURE IS REPEATED
                                        ;*FOR THE OTHER BYTE,  AFTER THIS, WRONG PARITY IS WRITTEN
                                        ;*FOR ALL 8 BIT COMBINATIONS IN BOTH THE LOW AND HIGH
                                        ;*BYTF SIMULTANEOUSLY FOR ONE LOC,  AFTER EACH DATA PATTERN
                                        ;*IS WRITTEN (R0 CONTAINS DATA PATTERN) A TRAP IS FORCED
                                        ;*AND THE PROGRAM CHECKS THAT THE TRAP WAS FROM BOTH HIGH
                                        ;*& LOW BYTES,
                                        ;*   FOLLOWING THIS ALL 8 BIT DATA PATTERNS FOR BOTH THE
                                        ;*HIGH & LOW BYTE ARE WRITTEN WITH GOOD PARITY IN ONE
                                        ;*CACHE LOC,  THE LOCATION IS REFERENCED AND ANY DATA
                                        ;*PARITY ERROR IS REPORTED,

                                        ;;*****************************************************************
010230  012737  000214  177746  TST14:  MOV     #214,@#CCR      ;TURN CACHE OFF FOR SCOPE
010236  000004                          SCOPE
010240  012737  012000  001234          MOV     #TST15,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
010246  012737  010350  000114          MOV     #T08L01,@#PVEC  ;SET UP PARITY TRAP HANDLER
010254  012700  062000                  MOV     #BUFH,R0        ;GET TEST ADDRESS
010260  005001                          CLR     R1              ;INIT FLAG TO INDIC, TESTING LOW BYTE
010262  005037  001166          T08L06: CLR     @#$REG4         ;SAVE DATA IF ERROR
010266  005037  001160                  CLR     @#$REG1         ;SAVE ADDRESS IF ERROR
010272  010037  001162                  MOV     R0,@#$REG2      ;SAVE ADDRESS IF ERROR
010276  012737  000204  177746          MOV     #204,@#CCR      ;TURN ON HALF OF CACHE
010304  005737  062000                  TST     @#BUFH          ;PUT LOC IN CACHE
010310  052737  000100  177746          BIS     #100,@#CCR      ;ENABLE WRITE WRONG PARITY
010316  112710  000000                  MOVB    #0,(R0)         ;DO DATOB TO LOC & WWP
010322  042737  000100  177746          BIC     #100,@#CCR      ;WWP OFF
010330  005737  062000                  TST     @#BUFH          ;FORCE PARITY TRAP
010334  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF

010342  104035                          ERROR   35              ;ERROR: TEST OF DATA PARITY GENERATOR/CKER FAILED
                                                                ;       DID NOT GET PARITY TRAP WHEN WROTE WRONG PARITY
```

```
010344  000137  011052                  JMP     T08L02          ;GO TO NEXT TEST

010350  012737  000214  177746  T08L01: MOV     #214,@#CCR      ;CACHE OFF

010356  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
010360  076600                          MED                     ;GET CONTENTS OF LOG REG
010362  000022                          .WORD   RLOG
010364  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
010370  076600                          MED                     ;UNLOCK ERROR LOG
010372  000722                          .WORD   WLOG
010374  012600                          MOV     (SP)+,R0        ;RESTORE R0

010376  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
010400  005701                          TST     R1              ;TESTING HIGH BYTE?
010402  001013                          BNE     T08L03          ;BRANCH IF YES
010404  032737  000100  177744          BIT     #100,@#EREG     ;WAS TRAP FROM LOW BYTE?
010412  001022                          BNE     T08L04          ;BRANCH IF YES

010414  076600                          MED                     ;GET LOG INFORMATION
010416  000106                          .WORD   CDL
010420  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
010424  104036                          ERROR   36              ;ERROR: TEST OF DATA PARITY GENERATOR/CKER FAILED
                                                                ;       DID NOT GET PARITY TRAP FROM LOW BYTE WHEN WWP
010426  000137  011052                  JMP     T08L02          ;GO TO NEXT TEST

010432  032737  000200  177744  T08L03: BIT     #200,@#EREG     ;WAS TRAP FROM HIGH BYTE?
010440  001012                          BNE     T08L05          ;BRANCH IF YES TO CONTINUE TEST
010442  076600                          MED                     ;GET LOG INFORMATION
010444  000106                          .WORD   CDH
010446  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
010452  104037                          ERROR   37              ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                ;       DID NOT GET PARITY TRAP FROM HIGH BYTE WHEN WWP
010454  000137  011052                  JMP     T08L02          ;GO TO NEXT TEST

010460  005200                  T08L04: INC     R0              ;TEST HIGH BYTE
010462  005201                          INC     R1              ;SET FLAG INDICATING HIGH BYTE TEST
010464  000676                          BR      T08L06          ;GO TEST IT

010466  012737  010546  000114  T08L05: MOV     #T08L07,@#PVEC  ;SET UP PARITY TRAP HANDLER
010474  012737  062000  001162          MOV     #BUFH,@#$REG2   ;SAVE ADDRESS IF ERROR
010502  005000                          CLR     R0              ;INIT, TEST DATA REG
010504  010037  001166          T08L10: MOV     R0,$REG4        ;SAVE DATA IF ERROR
010510  012737  000304  177746          MOV     #304,@#CCR      ;TURN HALF OF CACHE ON & WWP
010516  010037  062000                  MOV     R0,@#BUFH       ;GENERATE BAD PARITY AND WRITE IN CACHE
010522  042737  000100  177746          BIC     #100,@#CCR      ;WWP OFF
010530  005737  062000                  TST     @#BUFH          ;FORCE PARITY TRAP

010534  012737  000214  177746          MOV     #214,@#CCR      ;TURN CACHE OFF FOR ERROR
010542  104035                          ERROR   35              ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                ;       NO PARITY TRAP WHEN WROTE WRONG PARITY
010544  000542                          BR      T08L07          ;GO TO NEXT TEST

010546  012737  000214  177746  T08L07: MOV     #214,@#CCR      ;TURN CACHE OFF AFTER TRAP

010554  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
010556  076600                          MED                     ;GET CONTENTS OF LOG REG
```

```
010560  000022                            .WORD   RLOG
010562  052700  100001                    BIS     #100001,R0        ;ENABLE ERROR LOG & LOG FIRST MODE
010566  076600                            MED                       ;UNLOCK ERROR LOG
010570  000222                            .WORD   WLOG
010572  012600                            MOV     (SP)+,R0          ;RESTORE R0

010574  022626                            CMP     (SP)+,(SP)+       ;RESTORE STACK
010576  032737  000100  177744            BIT     #100,@#EREG       ;TRAP FROM LOW BYTE?
010604  001011                            BNE     T08L09            ;BRANCH IF YES

010606  076600                            MED                       ;GET LOG INFORMATION
010610  000106                            .WORD   CDL
010612  010037  001164                    MOV     R0,$REG3          ;SAVE INFORMATION
010616  012737  010504  001110            MOV     #T08L10,@#$LPERR  ;INIT FOR ERROR LOOP
010624  104036                            ERROR   36                ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                    ;       NO PARITY TRAP FROM LOW BYTE WHEN WWP
010626  000511                            BR      T08L02            ;GO TO END OF TEST

010630  032737  000200  177744  T08L09:   BIT     #200,@#EREG       ;TRAP FROM HIGH BYTE?
010636  001011                            BNE     T08L11            ;BRANCH IF YES

010640  076600                            MED                       ;GET LOG INFORMATION
010642  000106                            .WORD   CDH
010644  010037  001164                    MOV     R0,$REG3          ;SAVE INFORMATION
010650  012737  010504  001110            MOV     #T08L10,@#$LPERR  ;INIT FOR ERROR LOOP
010656  104037                            ERROR   37                ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                    ;       NO PARITY TRAP FROM HIGH BYTE WHEN WWP
010660  000474                            BR      T08L02            ;GO TO NEXT TEST

010662  022700  177777          T08L11:   CMP     #177777,R0        ;ALL WRITE WRONG PARITY PATTERNS CKED?
010666  001403                            BEQ     T08L12            ;BRANCH IF YES
010670  062700  000401                    ADD     #401,R0           ;GENERATE DATA FOR HIGH AND LOW BYTE
010674  000703                            BR      T08L10            ;GO TEST IT

010676  012737  010740  000114  T08L12:   MOV     #T08L13,@#PVEC    ;SET UP FOR PARITY ERRORS
010704  005000                            CLR     R0                ;INIT TEST DATA REG
010706  012737  000204  177746  T08L14:   MOV     #204,@#CCR        ;TURN HALF OF CACHE ON
010714  010037  062000                    MOV     R0,@#BUFH         ;GEN PARITY AND STORE IN CACHE
010720  005737  062000                    TST     @#BUFH            ;TEST PARITY
010724  022700  177777          T08L16:   CMP     #177777,R0        ;ALL GOOD PARITY PATTERNS CKED?
010730  001450                            BEQ     T08L02            ;BRANCH YES TO END OF TEST
010732  062700  000401                    ADD     #401,R0           ;GENERATE DATA FOR HIGH & LOW BYTE
010736  000763                            BR      T08L14            ;TEST IT

010740  052737  000014  177746  T08L13:   BIS     #14,@#CCR         ;TURN CACHE OFF

010746  010046                            MOV     R0,-(SP)          ;SAVE R0 FOR MED INST
010750  076600                            MED                       ;GET CONTENTS OF LOG REG
010752  000022                            .WORD   RLOG
010754  052700  100001                    BIS     #100001,R0        ;ENABLE ERROR LOG & LOG FIRST MODE
010760  076600                            MED                       ;UNLOCK ERROR LOG
010762  000222                            .WORD   WLOG
010764  012600                            MOV     (SP)+,R0          ;RESTORE R0

010766  022626                            CMP     (SP)+,(SP)+       ;RESTORE STACK
010770  010037  001166                    MOV     R0,$REG4          ;SAVE GOOD DATA
```

```
010774  076600                            MED                       ;GET LOG INFORMATION
010776  000106                            .WORD   RDAT
011000  010037  001164                    MOV     R0,$REG3          ;SAVE INFORMATION
011004  013700  001166                    MOV     $REG4,R0          ;RESTORE R0
011010  032737  000100  177744            BIT     #100,@#EREG       ;PARITY ERROR LOW BYTE?
011016  001405                            BEQ     T08L15            ;BRANCH IF NO
011020  012737  010706  001110            MOV     #T08L14,@#$LPERR  ;INIT ERROR LOOP
011026  104040                            ERROR   40                ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                    ;       PARITY ERROR IN LOW BYTE
011030  000410                            BR      T08L02            ;GO TO END OF TEST

011032  032737  000200  177744  T08L15:   BIT     #200,@#EREG       ;PARITY ERROR HIGH BYTE?
011040  001731                            BEQ     T08L16            ;TEST NEXT PATTERN IF NO
011042  012737  010706  001110            MOV     #T08L14,@#$LPERR  ;INIT RETURN FOR LOOP ON ERROR
011050  104041                            ERROR   41                ;ERROR: TEST OF DATA PARITY GEN//CKER FAILED
                                                                    ;       PARITY ERROR IN HIGH BYTE

011052                          T08L02:

                                          ;RID CACHE OF BAD PARITY
011052  012737  000214  177746            MOV     #214,@#CCR        ;CACHE OFF IF ON
011060  004737  035134                    JSR     PC,$WEEP          ;GO PURGE CACHE


011064  012737  033142  000114            MOV     #UPERR,@#114      ;RESTORE UNEXPECTED PARITY ERROR HANDLER
011072  000137  012000                    JMP     @#TST15           ;GO TO NEXT TEST


        012000                  .=12000                             ;ADJUST ADDRESS SPACE FOR NEXT TEST



                                ;;*************************************************************
                                ;*TEST 15        TEST THE VALID BIT FOR LOW HALF OF CACHE
                                ;*
                                ;*   THE TEST OF THE VALID BIT IS NOT COMPLETE UNTIL THE
                                ;*VALID TEST FOR THE SECOND HALF OF CACHE IS RUN.  THIS
                                ;*IS THE FIRST TEST WHERE THIS ENTIRE HALF OF CACHE ADDRESSES ARE
                                ;*EXERCISED.
                                ;*   DURING THE ENTIRE TEST ONLY ONE TAG AND DATA VALUE IS
                                ;*USED.  INITIALLY, THE ENTIRE HALF OF CACHE WHICH IS
                                ;*ENABLED (FORCE MISS OFF) IS WRITTEN AND CHECKED THAT ALL
                                ;*ITS ADDRESSES CAN BE MADE HITS.  FOLLOWING THIS, A WRITE/
                                ;*READ PROCEDURE IS DONE WHICH VERIFIES THAT THE LOCATIONS
                                ;*CAN BE VALIDATED/INVALIDATED AND THAT THERE IS NO DUAL
                                ;*ADDRESSING PROBLEM FOR THE V BIT.  FIRST THE VALID BIT
                                ;*IS SET FOR HALF OF CACHE, THEN STARTING AT THE LOWEST
                                ;*HALF CACHE ADDRESS, EACH LOC IS TESTED TO BE A HIT (VALID
                                ;*SET) AND THEN INVALIDATED VIA WRITING WRONG PARITY AND
                                ;*FORCING A TRAP.  THIS IS DONE INCREASING THE ADDRESS
                                ;*UNTIL HALF OF CACHE IS READ AND WRITTEN.  NEXT, STARTING
                                ;*AT THE HIGH HALF CACHE ADDRESS, EACH LOC IS READ, TESTED
                                ;*TO BE A MISS (VALID=0) AND THEN WRITTEN TO SET THE VALID
                                ;*BIT,  THIS IS DONE, DECREASING THE ADDRESS EACH TIME,
                                ;*TILL THE LOW ADDRESS IS REACHED.  THIS PROCEDURE IS THEN
                                ;*REPEATED FOR A SECOND PASS WITH THE PATTERN REVERSED.
```

```
                                        ;*(I.E. STARTING WITH ALL LOC INVALIDATED AND THEN READING
                                        ;*AND WRITING THE V BIT.)
                                        ;*
                                        ;*R0 CONTAINS THE CACHE ADDRESS BEING TESTED.

                                        ;*NOTE:TEST FOR DUAL ADDRESSING FOR LOCATIONS WHICH OVERLAP
                                        ;*      THE PARITY TRAP ADDRESSES 114,116 IS NOT DONE

                                        ;;*****************************************************************
  012000  012737  000214  177746  TST15: MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
  012006  000004                         SCOPE
  012010  012737  012734  001234         MOV     #TST16,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
  012016  012706  020000                 MOV     #20000,SP       ;ADJUST STACK FOR ADDRESSES OUT OF TEST AREA
  012022  012737  000210  177746         MOV     #210,@#CCR      ;HALF CACHE ON
  012030  012700  060000                 MOV     #BUFL,R0        ;INIT STARTING ADDRESS
  012034  012701  001000                 MOV     #1000,R1        ;INIT COUNT FOR 1/2 K
  012040  005020                  1$:    CLR     (R0)+           ;WRITE CACHE
  012042  077102                         SOB     R1,1$           ;LOOP TILL HALF CACHE WRITTEN
  012044  005740                  T24L20: TST    -(R0)           ;SEE IF DATA IN CACHE
  012046  033727  177752  000004         BIT     @#HMR,#HMR2     ;HIT? (VALID BIT SET?)
  012054  001002                         BNE     T24L19          ;BRANCH IF YES
  012056  000137  012624                 JMP     T24L01          ;REPORT ERROR
  012062  020027  060000          T24L19: CMP    R0,#BUFL        ;HALF CACHE TESTED?
  012066  001366                         BNE     T24L20          ;BRANCH IF NO


  012070  012737  012154  000114         MOV     #T24L02,@#PVEC  ;SET UP PARITY HANDLER

  012076  020027  060114          T24L05: CMP    R0,#BUFL!114    ;TESTING PARITY AREA?
  012102  001412                         BEQ     T24L22          ;DON'T TEST ADDRESS IF YES
  012104  020027  060116                 CMP     R0,#BUFL!116    ;TESTING PARITY AREA?
  012110  001407                         BEQ     T24L22          ;DON'T TEST ADDRESS IF YES
  012112  005710                         TST     (R0)            ;SEE IF VALID BIT SET
  012114  033727  177752  000004         BIT     @#HMR,#HMR2     ;HIT? (VALID BIT SET?)
  012122  001002                         BNE     T24L22          ;BRANCH IF YES
  012124  000137  012646                 JMP     T24L03          ;REPORT ERROR

  012130  012737  000310  177746  T24L22: MOV    #310,@#CCR      ;CACHE ON IF OFF AND WRITE WRONG PARITY
  012136  005010                         CLR     (R0)            ;WRITE LOC WITH WRONG PARITY
  012140  012737  000210  177746         MOV     #210,@#CCR      ;WWP OFF
  012146  005710                         TST     (R0)            ;FORCE PARITY TRAP
  012150  000137  012670                 JMP     T24L04          ;REPORT ERROR IF DID NOT TRAP

  012154                          T24L02:

  012154  010046                         MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
  012156  076600                         MED                     ;GET CONTENTS OF LOG REG
  012160  000022                         .WORD   RLOG
  012162  052700  100001                 BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
  012166  076600                         MED                     ;UNLOCK ERROR LOG
  012170  000222                         .WORD   WLOG
  012172  012600                         MOV     (SP)+,R0        ;RESTORE R0

  012174  062700  000002                 ADD     #2,R0           ;LOOK AT NEXT ADDR.
  012200  062706  000004                 ADD     #4,SP           ;RESTORE STACK
  012204  020027  062000                 CMP     R0,#BUFL+2000   ;HALF ADDRESSES TESTED?
```

```
  012210  001332                         BNE     T24L05          ;BRANCH IF NO

  012212  012737  033142  000114         MOV     #UPERR,@#PVEC   ;RESTORE UNEXP. PARITY ERROR HANDLER
  012220  005740                  1$:    TST     -(R0)           ;WAS LOC INVALIDATED?
  012222  033727  177752  000004         BIT     @#HMR,#HMR2     ;LOC A MISS? (INVALIDATED?)
  012230  001402                         BEQ     2$              ;BRANCH IF YES
  012232  000137  012712                 JMP     T24L06          ;REPORT ERROR
  012236  005010                  2$:    CLR     (R0)            ;WRITE LOC
  012240  020027  060000                 CMP     R0,#BUFL        ;AT LAST LOC?
  012244  001365                         BNE     1$              ;BRANCH IF NO

                                         ;NOW WRITE/READ VALID BIT WITH PATTERN REVERSED

  012246  012737  012316  000114  T24L10: MOV    #T24L07,@#PVEC  ;SET UP FOR PARITY TRAP
  012254  012700  061776                 MOV     #BUFL+1776,R0   ;INIT TEST ADDR.
  012260  012737  000310  177746  T24L08: MOV    #310,@#CCR      ;WRITE WRONG PARITY & CACHE ON
  012266  005010                         CLR     (R0)            ;WRITE WRONG PARITY
  012270  012737  000210  177746         MOV     #210,@#CCR      ;WWP OFF
  012276  005710                         TST     (R0)            ;FORCE TRAP
  012300  012737  000214  177746         MOV     #214,@#CCR      ;CACHE OFF
  012306  012737  012260  001110         MOV     #T24L08,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
  012314  000570                         BR      T24L15          ;REPORT ERROR IF DID NOT TRAP

  012316                          T24L07:

  012316  010046                         MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
  012320  076600                         MED                     ;GET CONTENTS OF LOG REG
  012322  000022                         .WORD   RLOG
  012324  052700  100001                 BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
  012330  076600                         MED                     ;UNLOCK ERROR LOG
  012332  000222                         .WORD   WLOG
  012334  012600                         MOV     (SP)+,R0        ;RESTORE R0

  012336  162700  000002                 SUB     #2,R0           ;LOOK AT NEXT ADDRESS
  012342  062706  000004                 ADD     #4,SP           ;ADJUST STACK
  012346  020027  057776                 CMP     R0,#BUFL-2      ;HALF CACHE WRITTEN?
  012352  001342                         BNE     T24L08          ;BRANCH IF NO

  012354  012737  033142  000114         MOV     #UPERR,@#PVEC
  012362  062700  000002          T24L12: ADD    #2,R0           ;ADJUST ADDRESS
  012366  005710                         TST     (R0)            ;READ LOC
  012370  033727  177752  000004         BIT     @#HMR,#HMR2     ;MISS? (LOC INVALIDATED?)
  012376  001407                         BEQ     T24L09          ;BRANCH IF YES
  012400  012737  000214  177746         MOV     #214,@#CCR      ;CACHE OFF
  012406  012737  012246  001110         MOV     #T24L10,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
  012414  000536                         BR      T24L06          ;REPORT ERROR

  012416  005010                  T24L09: CLR    (R0)            ;WRITE LOC
  012420  020027  061776                 CMP     R0,#BUFL+1776   ;HALF CACHE WRITTEN?
  012424  001356                         BNE     T24L12          ;BRANCH IF NO

                                         ;NOW READ LOC TO SEE IF VALID STILL SET

  012426  012737  012536  000114         MOV     #T24L16,@#PVEC  ;SET UP PARITY HANDLER
  012434  020027  060114          T24L17: CMP    R0,#BUFL!114    ;TESTING PARITY AREA?
  012440  001417                         BEQ     T24L13          ;DON'T TEST ADDRESS IF YES
```

```
 012442  020027  060116                         CMP     R0,#BUFL1116      ;TESTING PARITY AREA?
 012446  001414                                 BEQ     T24L13            ;DON'T TEST ADDRESS IF YES

 012450  005710                                 TST     (R0)              ;LOC IN CACHE?
 012452  033727  177752  000004                 BIT     @#HMR,#HMR2       ;HIT?
 012460  001007                                 BNE     T24L13            ;BRANCH IF YES
 012462  012737  000214  177746                 MOV     #214,@#CCR        ;CACHE OFF
 012470  012737  012246  001110                 MOV     #T24L10,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
 012476  000466                                 BR      T24L14            ;REPORT ERROR

 012500  052737  000100  177746         T24L13: BIS     #100,@#CCR        ;SET WRITE WRONG PARITY
 012506  005010                                 CLR     (R0)              ;WRITE WRONG PARITY
 012514  012737  000210  177746                 MOV     #210,@#CCR        ;WWP OFF
 012516  005710                                 TST     (R0)              ;FORCE TRAP
 012520  012737  000214  177746                 MOV     #214,@#CCR        ;CACHE OFF
 012526  012737  012246  001110                 MOV     #T24L10,@#$LPERR
 012534  000460                                 BR      T24L15            ;REPORT ERROR

 012536  062706  000004                 T24L16: ADD     #4,SP             ;RESTORE STACK
 012542  162700  000002                         SUB     #2,R0             ;LOOK AT NEXT ADDR.

 012546  010046                                 MOV     R0,-(SP)          ;SAVE R0 FOR MED INST
 012550  076600                                 MED                       ;GET CONTENTS OF LOG REG
 012552  000022                                 .WORD   RLOG
 012554  052700  100001                         BIS     #100001,R0        ;ENABLE ERROR LOG & LOG FIRST MODE
 012560  076600                                 MED                       ;UNLOCK ERROR LOG
 012562  000222                                 .WORD   WLOG
 012564  012600                                 MOV     (SP)+,R0          ;RESTORE R0

 012566  020027  057776                         CMP     R0,#BUFL-2        ;ALL ADDR TESTED?
 012572  001320                                 BNE     T24L17            ;BRANCH IF NO

 012574                                 T24L18:


                                                ;RID CACHE OF BAD PARITY
 012574  012737  000214  177746                 MOV     #214,@#CCR        ;CACHE OFF IF ON
 012602  004737  035134                         JSR     PC,SWEEP          ;GO PURGE CACHE


 012606  012737  033142  000114                 MOV     #UPERR,@#PVEC
 012614  012706  001100                         MOV     #STACK,SP         ;RESTORE STACK
 012620  000137  012734                         JMP     @#TST16           ;GO TO NEXT TEST

 012624  012737  000214  177746         T24L01: MOV     #214,@#CCR        ;CACHE OFF
 012632  005037  001160                         CLR     $REG1             ;SAVE FAILING ADDR
 012636  010037  001162                         MOV     R0,$REG2          ;SAVE FAILING ADDR
 012642  104043                                 ERROR   43                ;ERROR: ADDRESS COULD NOT BE MADE A HIT
 012644  000753                                 BR      T24L18            ;GO TO END OF TEST

 012646  012737  000214  177746         T24L03: MOV     #214,@#CCR        ;CACHE OFF
 012654  005037  001160                 T24L14: CLR     $REG1             ;SAVE FAILING ADDRESS
 012660  010037  001162                         MOV     R0,$REG2          ;SAVE FAILING ADDRESS
 012664  104111                                 ERROR   111               ;ERROR: TEST OF VALID BIT FAILED
                                                                          ;       LOC COULD NOT BE MADE A HIT
 012666  000742                                 BR      T24L18            ;GO TO END OF TEST

```

```
 012670  012737  000214  177746         T24L04: MOV     #214,@#CCR        ;CACHE OFF
 012676  005037  001160                 T24L15: CLR     $REG1             ;SAVE FAILING ADDRESS
 012702  010037  001162                         MOV     R0,$REG2          ;SAVE FAILING ADDRESS
 012706  104042                                 ERROR   42                ;ERROR: NO PARITY TRAP FROM LOC WRITTEN WITH WRONG PARIT
 012710  000731                                 BR      T24L18            ;GO TO END OF TEST

 012712  012737  000214  177746         T24L06: MOV     #214,@#CCR        ;CACHE OFF
 012720  005037  001160                         CLR     $REG1             ;SAVE FAILING ADDR
 012724  010037  001162                         MOV     R0,$REG2          ;SAVE FAILING ADDR
 012730  104112                                 ERROR   112               ;ERROR: TEST OF VALID BIT FAILED
                                                                          ;       LOCATION NOT INVALIDATED BY PARITY TRAP
 012732  000720                                 BR      T24L18            ;GO TO END OF TEST

                                        ;;*************************************************************
                                        ;*TEST 16       TEST TAG PARITY BIT FOR LOW CACHE ADDRESSES
                                        ;*
                                        ;*
                                        ;*   THE TEST OF THE TAG PARITY BIT IS NOT COMPLETE UNTIL
                                        ;*THE TAG P BIT TEST FOR THE SECOND HALF OF CACHE AND THE
                                        ;*MSB ADDRESS (A10) TO CACHE TAG FIELD TEST ARE RUN.  TWO
                                        ;*TAG ADDRESSES ARE USED TO GENERATE A PARITY BIT OF 1 AND
                                        ;*0.  THE FIRST ADDRESS IS CHOSEN FROM A TEST BUFFER AREA
                                        ;*AND THE SECOND IS CHOSEN TO LIE 1K AWAY.  A WRITE/READ
                                        ;*PROCEDURE IS DONE WHICH CHECKS THE P BIT AND DUAL ADD-
                                        ;*RESSING FOR HALF OF CACHE.  INITIALLY THE P BIT IS WRITTEN
                                        ;*WITH ONE PARITY PATTERN IN HALF OF CACHE.  THEN STARTING
                                        ;*AT THE LOW HALF CACHE ADDRESS, THE LOC IS READ AND THEN
                                        ;*WRITTEN WITH THE OPPOSITE PARITY.  THIS IS SEQUENTIALLY
                                        ;*REPEATED WITH INCREASING ADDRESSES UNTIL THE HIGH HALF
                                        ;*CACHE ADDRESS IS REACHED.  THEN STARTING AT THE HIGH ADDR,
                                        ;*THE SECOND PARITY PATTERN IS READ AND THE LOC IS REWRITTEN
                                        ;*WITH THE FIRST.  THIS IS SEQUENTIALLY REPEATED, DECREASING
                                        ;*THE ADDRESS, UNTIL THE LOW HALF CACHE ADDRESS IS REACHED.
                                        ;*A SECOND PASS IS THEN MADE WITH THE PARITY PATTERN RE-
                                        ;*VERSED.  A PARITY ERROR HANDLER IS SETUP TO DETECT PARITY
                                        ;*ERRORS.  ALSO, LOCS WHICH SHOULD BE HITS ARE CHECKED FOR
                                        ;*AND REPORTED IF NO HIT OCCURRED.
                                        ;*
                                        ;*R0, R1 CONTAIN ADDRESSES TO GENERATE COMPLIMENTARY TAG
                                        ;*PARITY BITS.

                                        ;;*************************************************************
 012734  012737  000214  177746         TST16:  MOV     #214,@#CCR        ;CACHE OFF FOR SCOPE
 012742  000004                                 SCOPE
 012744  012737  013406  001234                 MOV     #TST17,$KTST      ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 012752  012737  013126  000114                 MOV     #T11L01,@#PVEC    ;SET UP FOR PARITY ERRORS
 012760  005003                                 CLR     R3                ;INIT FLAG=FIRST PASS
 012762  012700  060000                         MOV     #BUFL,R0          ;SET UP ADDR. FOR FIRST PASS
 012766  012737  000210  177746                 MOV     #210,@#CCR        ;TURN HALF CACHE ON
 012774  012701  001000                 T11L02: MOV     #1000,R1          ;INIT COUNTER
 013000  005720                         1$:     TST     (R0)+             ;PUT PARITY PATTERN IN TAG FIELD
 013002  077102                                 SOB     R1,1$             ;LOAD HALF OF CACHE

 013004  012701  001000                         MOV     #1000,R1          ;INIT. COUNTER
 013010  012700  060000                         MOV     #BUFL,R0          ;SET UP ADDR. FOR FIRST PASS
 013014  012702  054000                         MOV     #BUFL-4000,R2     ;SET UP ADDR. FOR FIRST PASS
```

```
013020  005703                          TST     R3              ;FIRST PASS?
013022  001404                          BEQ     T11L03          ;BRANCH IF YES
013024  012700  054000                  MOV     #BUFL-4000,R0   ;SET UP ADDR. FOR SECOND PASS
013030  012702  060000                  MOV     #BUFL,R2        ;SET UP ADDR. FOR SECOND PASS
013034  005720                  T11L03: TST     (R0)+           ;READ CACHE TO SEE IF PARITY OK; NO-TRAPS
013036  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS ADDRESS A HIT?
013044  001533                          BEQ     T11L04          ;BRANCH TO ERROR IF NO
013046  005722                          TST     (R2)+           ;WRITE DIFFERENT PARITY PATTERN IN TAG FIELD
013050  077107                          SOB     R1,T11L03       ;LOOK AT HALF OF CACHE

013052  012701  001000                  MOV     #1000,R1        ;INIT COUNTER
013056  005742                  T11L11: TST     -(R2)           ;READ SECOND PARITY PATTERN
013060  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS ADDRESS A HIT?
013066  001532                          BEQ     T11L05          ;BRANCH IF NO TO ERROR
013070  005740                          TST     -(R0)           ;PUT NEW PARITY PATTERN IN TAG
013072  077107                          SOB     R1,T11L11       ;LOOK AT HALF OF CACHE

013074  005703                          TST     R3              ;FIRST PASS?
013076  001140                          BNE     T11L06          ;NO GO TO END OF TEST
013100  052703  000001                  BIS     #1,R3           ;SET FLAG TO INDIC. SECOND PASS
013104  012737  000210  177746  T11L12: MOV     #210,@#CCR      ;HALF CACHE ON IF OFF
013112  012737  013104  001110          MOV     #T11L12,@#$LPERR  ;SETUP RETURN FOR ERROR IF ONE OCCURS
013120  012700  054000                  MOV     #BUFL-4000,R0   ;SET UP FOR SECOND PASS.
013124  000723                          BR      T11L02          ;GO TEST SECOND PASS

013126                          T11L01:

                                        ;RID CACHE OF BAD PARITY
013126  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
013134  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE



013140  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
013142  076600                          MED                     ;GET CONTENTS OF LOG REG
013144  000022                          .WORD   RLOG
013146  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
013152  076600                          MED                     ;UNLOCK ERROR LOG
013154  000222                          .WORD   WLOG
013156  012600                          MOV     (SP)+,R0        ;RESTORE R0

013160  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
013162  000101                          .WORD   RSER
013164  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
013166  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
013172  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
013176  076600                          MED                     ;GET LOG INFORMATION
013200  000102                          .WORD   LOADD
013202  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
013206  076600                          MED                     ;GET LOG INFORMATION
013210  000100                          .WORD   RJAM
013212  032700  000400                  BIT     #400,R0         ;ERROR IN BACKING STORE?
013216  001413                          BEQ     T11L07          ;BRANCH IF NO
013220  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 WHERE ERROR OCCURRED
013224  162737  000002  001166          SUB     #2,$REG4        ;SAVE PC WHERE ERROR OCCURRED
013232  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
```

```
013234  104001                          ERROR   1               ;ERROR; UNEXPECTED PARITY ERROR IN BACKING STORE
013236  000460                          BR      T11L06          ;GO TO NEXT TEST

013240  022626                  T11L07: CMP     (SP)+,(SP)+     ;RESTORE STACK
013242  032737  000040  177744          BIT     #40,@#EREG      ;ERROR IN TAG?
013250  001411                          BEQ     T11L08          ;BRANCH NO
013252  076600                          MED                     ;GET TAG LOG INFO.
013254  000107                          .WORD   RTAG
013256  000300                          SWAB    R0              ;PUT TAG IN LOW BYTE
013260  042700  177400                  BIC     #177400,R0      ;LOOK AT TAG ONLY
013264  010037  001164                  MOV     R0,$REG3        ;SAVE BAD DATA
013270  104045                          ERROR   45              ;ERROR; TAG PARITY ERROR WHEN TESTING TAG P BIT
013272  000442                          BR      T11L06          ;GO TO NEXT TEST

013274  032737  000100  177744  T11L08: BIT     #100,@#EREG     ;ERROR IN LOW BYTE?
013302  001406                          BEQ     T11L09          ;BRANCH IF NO

013304  076600                          MED                     ;GET LOG INFORMATION
013306  000106                          .WORD   CDL
013310  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
013314  104046                          ERROR   46              ;ERROR; LOW BYTE PARITY ERROR WHEN TESTING TAG P BIT
013316  000430                          BR      T11L06          ;NEXT TEST

013320                          T11L09:
013320  076600                          MED                     ;GET LOG INFORMATION
013322  000106                          .WORD   CDH
013324  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
013330  104047                          ERROR   47              ;ERROR; HIGH BYTE PARITY ERROR WHEN TESTING TAG P BIT
013332  000422                          BR      T11L06          ;NEXT TEST

013334  052737  000014  177746  T11L04: BIS     #14,@#CCR       ;CACHE OFF
013342  162700  000002                  SUB     #2,R0           ;GET BAD ADDRESS
013346  010037  001162                  MOV     R0,$REG2        ;SAVE BAD ADDRESS
013352  000407                          BR      T11L10          ;REPORT ERROR
013354  052737  000014  177746  T11L05: BIS     #14,@#CCR       ;CACHE OFF
013362  010237  001162                  MOV     R2,$REG2        ;SAVE BAD ADDRESS
013366  062702  000002                  ADD     #2,R2           ;RESTORE R2 TO FAILING ADDR.+2
013372  005037  001160          T11L10: CLR     $REG1           ;SAVE BAD ADDRESS
013376  104043                          ERROR   43              ;ERROR; ADDRESS COULD NOT BE MADE A HIT

013400  012737  033142  000114  T11L06: MOV     #UPERR,@#PVEC   ;RESTORE PARITY TRAP HANDLER

                                ;;*************************************************************
                                ;*TEST 17        TEST DATA PARITY BITS FOR LOW CACHE
                                ;*
                                ;*   THE TEST OF THE DATA PARITY BITS ARE NOT COMPLETE
                                ;*UNTIL THE DATA P BIT TEST FOR THE SECOND HALF OF CACHE
                                ;*AND THE MSB ADDRESS (A10) TO CACHE DATA FIELD ARE RUN.
                                ;*A WRITE/READ PROCEDURE IS DONE WHICH SIMULTANEOUSLY
                                ;*CHECKS THE DATA P BIT FOR BOTH BYTES AND DUAL ADDRESSING
                                ;*IN HALF OF CACHE FOR IT.  INITIALLY THE P BIT IS WRITTEN
                                ;*WITH ONE PARITY PATTERN IN HALF OF CACHE.  THEN STARTING
                                ;*AT THE LOW HALF CACHE ADDRESS, THE LOC IS READ AND THEN
                                ;*WRITTEN WITH THE OPPOSITE PARITY.  THIS IS  SEQUEN-
                                ;*TIALLY REPEATED WITH INCREASING ADDRESSES UNTIL THE HIGH
                                ;*HALF CACHE ADDRESS IS REACHED.  THEN STARTING AT THE
                                ;*HIGH ADDR. THE SECOND PARITY PATTERN IS READ AND THE LOC
```

```
                                              ;*IS REWRITTEN WITH THE FIRST.  THIS IS SEQUENTIALLY RE-
                                              ;*PEATED DECREASING THE ADDRESS UNTIL THE LOW HALF CACHE
                                              ;*ADDRESS IS REACHED.  A SECOND PASS IS THEN MADE WITH
                                              ;*THE PARITY PATTERN REVERSED.  A PARITY ERROR HANDLER IS
                                              ;*SETUP TO DETECT PARITY ERRORS.  ALSO, LOCS WHICH SHOULD
                                              ;*BE HITS ARE CHECKED FOR AND REPORTED IF NO HIT OCCURRED.
                                              ;*
                                              ;*R0, R1 CONTAIN DATA WHICH GENERATE OPPOSITE PARITY.  R3
                                              ;*INDICATES WHICH PASS IS BEING DONE.

                                              ;;*****************************************************************
013406  012737  000214  177746  TST17:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
013414  000004                          SCOPE
013416  012737  014100  001234          MOV     #TST20,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
013424  012737  013646  000114          MOV     #T12L01,@#PVEC  ;SET UP PARITY ERROR HANDLER
013432  005003                          CLR     R3              ;INIT FLAG FOR FIRST PASS
013434  005000                          CLR     R0              ;SET UP PARITY PATTERN A FOR FIRST PASS
013436  012737  000210  177746  T12L02: MOV     #210,@#CCR      ;HALF CACHE ON
013444  012701  001000                  MOV     #1000,R1        ;INIT ADDR. COUNTER
013450  012705  060000                  MOV     #BUFL,R5        ;INIT. TEST ADDRESS
013454  010025                  1$:     MOV     R0,(R5)+        ;WRITE DATA PARITY PATTERN
013456  077102                          SOB     R1,1$           ;HALF ADDR. WRITTEN? BRANCH IF NO

013460  012701  001000                  MOV     #1000,R1        ;INIT ADDR. COUNTER
013464  012705  060000                  MOV     #BUFL,R5        ;INIT. TEST ADDR
013470  012700  000401                  MOV     #401,R0         ;SET UP PATTERN B FOR FIRST PASS
013474  005703                          TST     R3              ;FIRST PASS?
013476  001401                          BEQ     2$              ;BRANCH IF YES
013500  005000                          CLR     R0              ;SET UP PARITY PATTERN A FOR SECOND PASS
013502  005715                  2$:     TST     (R5)            ;SEE IF PARITY UNCHANGED
013504  033727  177752  000004          BIT     @#HMR,#HMR2     ;DATA FROM CACHE?
013512  001444                          BEQ     T12L07          ;BRANCH TO ERROR IF NO
013514  010025                          MOV     R0,(R5)+        ;WRITE NEW DATA PARITY PATTERN
013516  077107                          SOB     R1,2$           ;HALF ADDR. SPACE EXAMINED & WRITTEN?

013520  012701  001000                  MOV     #1000,R1        ;INIT ADDR. COUNTER
013524  005000                          CLR     R0              ;SET UP PARITY PATTERN A FOR FIRST PASS
013526  005703                          TST     R3              ;FIRST PASS?
013530  001402                          BEQ     T12L06          ;BRANCH IF YES
013532  012700  000401                  MOV     #401,R0         ;SET UP PARITY PATTERN B FOR SECOND PASS
013536  012737  000210  177746  T12L06: MOV     #210,@#CCR      ;HALF CACHE ON IF OFF FROM ERROR
013544  005745                  1$:     TST     -(R5)           ;SEE IF PARITY UNCHANGED
013546  033727  177752  000004          BIT     @#HMR,#HMR2     ;DATA FROM CACHE
013554  001423                          BEQ     T12L07          ;BRANCH IF NO TO ERROR
013556  010015                          MOV     R0,(R5)         ;WRITE NEW PARITY PATTERN IN CACHE
013560  077107                          SOB     R1,1$           ;HALF OF ADDRESS SPACE READ & WRITTEN? BRANCH IF NO

013562  005703                          TST     R3              ;SECOND PASS?
013564  001010                          BNE     T12L08          ;GO TO END OF TEST IF YES
013566  012700  000401          T12L13: MOV     #401,R0         ;SET UP PARITY PATTERN B FOR SECOND PASS
013572  052703  000001                  BIS     #1,R3           ;SET FLAG FOR PASS 2
013576  012737  013566  001110          MOV     #T12L13,@#$LPERR  ;INIT RETURN FOR ERROR LOOP IF ERROR OCCURS
013604  000714                          BR      T12L02          ;TEST DATA


013606  012737  033142  000114  T12L08: MOV     #UPERR,@#PVEC   ;RESTORE PARITY ERROR HANDLER
```

```
013614  052737  000014  177746          BIS     #14,@#CCR       ;CACHE OFF WHEN CROSS CACHE ADDRESS BOUNDARY
013622  000526                          BR      TST20           ;;GO TO NEXT TEST

013624  052737  000014  177746  T12L07: BIS     #14,@#CCR       ;CACHE OFF
013632  010537  001162                  MOV     R5,$REG2        ;SAVE BAD ADDRESS
013636  005037  001160                  CLR     $REG1           ;SAVE BAD ADDRESS
013642  104043                          ERROR   43              ;ERROR: ADDRESS COULD NOT BE MADE A HIT
013644  000760                          BR      T12L08          ;GO TO END OF TEST

013646  052737  000014  177746  T12L01: BIS     #14,@#CCR       ;CACHE OFF

013654  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
013656  076600                          MED                     ;GET CONTENTS OF LOG REG
013660  000022                          .WORD   RLOG
013662  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
013666  076600                          MED                     ;UNLOCK ERROR LOG
013670  000222                          .WORD   WLOG
013672  012600                          MOV     (SP)+,R0        ;RESTORE R0

013674  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
013676  000101                          .WORD   RSER
013700  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
013702  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
013706  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
013712  076600                          MED                     ;GET LOG INFORMATION
013714  000102                          .WORD   LOADD
013716  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
013722  032737  000040  177744          BIT     #40,@#EREG      ;ERROR IN TAG?
013730  001417                          BEQ     T12L09          ;BRANCH IF NO
013732  011637  001166                  MOV     (SP),$REG4      ;GET PC+2 OF ERROR
013736  162737  000002  001166          SUB     #2,$REG4        ;GET PC OF ERROR
013744  076600                          MED                     ;GET TAG LOG INFO.
013746  000107                          .WORD   RTAG
013750  000300                          SWAB    R0              ;PUT TAG IN LOW BYTE
013752  042700  177400                  BIC     #177400,R0      ;LOOK AT TAG ONLY
013756  010037  001164                  MOV     R0,$REG3        ;SAVE BAD DATA
013762  022626                          CMP     (SP)+,(SP)+     ;RESTORE THE STACK
013764  104002                          ERROR   2               ;ERROR: UNEXPECTED PARITY ERROR IN TAG FIELD
013766  000707                          BR      T12L08          ;GO TO END OF TEST

013770  022626                  T12L09: CMP     (SP)+,(SP)+     ;RESTORE STACK
013772  005037  001166                  CLR     $REG4           ;SAVE GOOD DATA
013776  005700                          TST     R0              ;WAS TEST DATA =0?
014000  001003                          BNE     T12L11          ;BRANCH IF NO
014002  012737  000401  001166          MOV     #401,$REG4      ;SAVE GOOD DATA
014010  032737  000200  177744  T12L11: BIT     #200,@#EREG     ;ERROR IN HIGH BYTE?
014016  001406                          BEQ     T12L12          ;BRANCH IF NO
014020  076600                          MED                     ;GET LOG INFORMATION
014022  000106                          .WORD   CDH
014024  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
014030  104050                          ERROR   50              ;ERROR: HIGH BYTE PARITY ERROR WHEN TESTING DATA P BITS
014032  000665                          BR      T12L08          ;GO TO END OF TEST

014034  032777  000100  163702  T12L12: BIT     #100,@EREG      ;ERROR IN LOW BYTE?
014042  001406                          BEQ     T12L14          ;BRANCH IF NO
014044  076600                          MED                     ;GET LOG INFORMATION
```

```
014046  000106                               .WORD   CDL
014050  010037  001164                       MOV     R0,$REG3        ;SAVE INFORMATION
014054  104051                               ERROR   51              ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA P BITS
014056  000653                               BR      T12L08          ;GO TO END OF TEST

014060  016637  177774  001164  T12L14: MOV     -4(SP),$REG3    ;GET PC+2 OF TRAP
014066  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF TRAP
014074  104001                               ERROR   1               ;ERROR: UNEXP. PARITY ERROR IN BACKING STORE
014076  000643                               BR      T12L08          ;GO TO END OF TEST



                                        ;;*******************************************************************
                                        ;*TEST 20        TEST THE VALID BIT FOR HIGH HALF OF CACHE
                                        ;*
                                        ;*   THE TEST OF THE VALID BIT IS NOT COMPLETE UNTIL THE
                                        ;*VALID TEST FOR THE SECOND HALF OF CACHE IS RUN.  THIS
                                        ;*IS THE FIRST TEST WHERE THIS ENTIRE HALF OF CACHE ADDRESSES ARE
                                        ;*EXERCISED.
                                        ;*   DURING THE ENTIRE TEST ONLY ONE TAG AND DATA VALUE IS
                                        ;*USED.  INITIALLY, THE ENTIRE HALF OF CACHE WHICH IS
                                        ;*ENABLED (FORCE MISS OFF) IS WRITTEN AND CHECKED THAT ALL
                                        ;*ITS ADDRESSES CAN BE MADE HITS.  FOLLOWING THIS, A WRITE/
                                        ;*READ PROCEDURE IS DONE WHICH VERIFIES THAT THE LOCATIONS
                                        ;*CAN BE VALIDATED/INVALIDATED AND THAT THERE IS NO DUAL
                                        ;*ADDRESSING PROBLEM FOR THE V BIT.  FIRST THE VALID BIT
                                        ;*IS SET FOR HALF OF CACHE, THEN STARTING AT THE LOWEST
                                        ;*HALF CACHE ADDRESS, EACH LOC IS TESTED TO BE A HIT (VALID
                                        ;*SET) AND THEN INVALIDATED VIA WRITING WRONG PARITY AND
                                        ;*FORCING A TRAP.  THIS IS DONE INCREASING THE ADDRESS
                                        ;*UNTIL HALF OF CACHE IS READ AND WRITTEN.  NEXT, STARTING
                                        ;*AT THE HIGH HALF CACHE ADDRESS, EACH LOC IS READ, TESTED
                                        ;*TO BE A MISS (VALID=0) AND THEN WRITTEN TO SET THE VALID
                                        ;*BIT.  THIS IS DONE, DECREASING THE ADDRESS EACH TIME,
                                        ;*TILL THE LOW ADDRESS IS REACHED.  THIS PROCEDURE IS THEN
                                        ;*REPEATED FOR A SECOND PASS WITH THE PATTERN REVERSED.
                                        ;*(I.E. STARTING WITH ALL LOC INVALIDATED AND THEN READING
                                        ;*AND WRITING THE V BIT.)
                                        ;*
                                        ;*R0 CONTAINS THE CACHE ADDRESS BEING TESTED.

                                        ;;*******************************************************************
014100  012737  000214  177746  TST20:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
014106  000004                          SCOPE
014110  012737  015000  001234          MOV     #TST21,BKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
014116  012737  000204  177746          MOV     #204,@#CCR      ;HALF CACHE ON
014124  012700  062000                  MOV     #BUFH,R0        ;INIT STARTING ADDRESS
014130  012701  001000                  MOV     #1000,R1        ;INIT COUNT FOR 1/2 K
014134  005020                  1$:     CLR     (R0)+           ;WRITE CACHE
014136  077102                          SOB     R1,1$           ;LOOP TILL HALF CACHE WRITTEN

014140  005740                  T24H20: TST     -(R0)           ;SEE IF DATA IN CACHE
014142  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT? (VALID BIT SET?)
014150  001002                          BNE     T24H19          ;BRANCH IF YES
014152  000137  014670                  JMP     T24H01          ;REPORT ERROR
014156  020027  062000          T24H19: CMP     R0,#BUFH        ;HALF CACHE TESTED?
```

```
014162  001366                          BNE     T24H20          ;BRANCH IF NO

014164  012737  014242  000114          MOV     #T24H02,@#PVEC  ;SET UP PARITY HANDLER
014172  012737  000204  177746  T24H21: MOV     #204,@#CCR      ;CACHE ON IF OFF
014200  005710                  T24H05: TST     (R0)            ;SEE IF VALID BIT SET
014202  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT? (VALID BIT SET?)
014210  001002                          BNE     T24H22          ;BRANCH IF YES
014212  000137  014712                  JMP     T24H03          ;REPORT ERROR

014216  012737  000304  177746  T24H22: MOV     #304,@#CCR      ;CACHE ON IF OFF AND WRITE WRONG PARITY
014224  005010                          CLR     (R0)            ;WRITE LOC WITH WRONG PARITY
014226  012737  000204  177746          MOV     #204,@#CCR      ;WWP OFF
014234  005710                          TST     (R0)            ;FORCE PARITY TRAP
014236  000137  014734                  JMP     T24H04          ;REPORT ERROR IF DID NOT TRAP

014242                          T24H02:

014242  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
014244  076600                          MED                     ;GET CONTENTS OF LOG REG
014246  000022                          .WORD   RLOG
014250  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
014254  076600                          MED                     ;UNLOCK ERROR LOG
014256  000222                          .WORD   WLOG
014260  012600                          MOV     (SP)+,R0        ;RESTORE R0

014262  062700  000002                  ADD     #2,R0           ;LOOK AT NEXT ADDR.
014266  062706  000004                  ADD     #4,SP           ;RESTORE STACK
014272  020027  064000                  CMP     R0,#BUFH+2000   ;HALF ADDRESSES TESTED?
014276  001340                          BNE     T24H05          ;BRANCH IF NO

014300  012737  033142  000114          MOV     #UPERR,@#PVEC   ;RESTORE UNEXP. PARITY ERROR HANDLER
014306  005740                  1$:     TST     -(R0)           ;WAS LOC INVALIDATED?
014310  033727  177752  000004          BIT     @#HMR,#HMR2     ;LOC A MISS? (INVALIDATED?)
014316  001402                          BEQ     2$              ;BRANCH IF YES
014320  000137  014756                  JMP     T24H06          ;REPORT ERROR
014324  005010                  2$:     CLR     (R0)            ;WRITE LOC
014326  020027  062000                  CMP     R0,#BUFH        ;AT LAST LOC?
014332  001365                          BNE     1$              ;BRANCH IF NO

                                        ;NOW WRITE/READ VALID BIT WITH PATTERN REVERSED

014334  012737  014404  000114  T24H10: MOV     #T24H07,@#PVEC  ;SET UP FOR PARITY TRAP
014342  012700  063776                  MOV     #BUFH+1776,R0   ;INIT TEST ADDR.
014346  012737  000304  177746  T24H08: MOV     #304,@#CCR      ;WRITE WRONG PARITY & CACHE ON
014354  005010                          CLR     (R0)            ;WRITE WRONG PARITY & CACHE ON
014356  012737  000204  177746          MOV     #204,@#CCR      ;WWP OFF
014364  005710                          TST     (R0)            ;FORCE TRAP
014366  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF
014374  012737  014346  001110          MOV     #T24H08,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
014402  000557                          BR      T24H15          ;REPORT ERROR IF DID NOT TRAP

014404                          T24H07:

014404  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
014406  076600                          MED                     ;GET CONTENTS OF LOG REG
014410  000022                          .WORD   RLOG
```

```
 014412  052700  100001                   BIS     #100001,R0       ;ENABLE ERROR LOG & LOG FIRST MODE
 014416  076600                           MED                      ;UNLOCK ERROR LOG
 014420  000222                           .WORD   WLOG
 014422  012600                           MOV     (SP)+,R0         ;RESTORE R0

 014424  162700  000002                   SUB     #2,R0            ;LOOK AT NEXT ADDRESS
 014430  062706  000004                   ADD     #4,SP            ;ADJUST STACK
 014434  020027  061776                   CMP     R0,#BUFH-2       ;HALF CACHE WRITTEN?
 014440  001342                           BNE     T24H08           ;BRANCH IF NO

 014442  012737  033142  000114           MOV     #UPERR,@#PVEC
 014450  062700  000002          T24H12:  ADD     #2,R0            ;ADJUST ADDRESS
 014454  005710                           TST     (R0)             ;READ LOC
 014456  033727  177752  000004           BIT     @#HMR,#HMR2      ;MISS? (LOC INVALIDATED?)
 014464  001407                           BEQ     T24H09           ;BRANCH IF YES
 014466  012737  000214  177746           MOV     #214,@#CCR       ;CACHE OFF
 014474  012737  014334  001110           MOV     #T24H10,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
 014502  000525                           BR      T24H06           ;REPORT ERROR

 014504  005010                  T24H09:  CLR     (R0)             ;WRITE LOC
 014506  020027  063776                   CMP     R0,#BUFH+1776    ;HALF CACHE WRITTEN?
 014512  001356                           BNE     T24H12           ;BRANCH IF NO

                                          ;NOW READ LOC TO SEE IF VALID STILL SET

 014514  012737  014610  000114           MOV     #T24H16,@#PVEC   ;SET UP PARITY HANDLER
 014522  005710                  T24H17:  TST     (R0)             ;LOC IN CACHE?
 014524  033727  177752  000004           BIT     @#HMR,#HMR2      ;HIT?
 014532  001007                           BNE     T24H11           ;BRANCH IF YES
 014534  012737  000214  177746           MOV     #214,@#CCR       ;CACHE OFF
 014542  012737  014334  001110           MOV     #T24H10,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
 014550  000463                           BR      T24H14           ;REPORT ERROR

 014552  052737  000100  177746  T24H13:  BIS     #100,@#CCR       ;SET WRITE WRONG PARITY
 014560  005010                           CLR     (R0)             ;WRITE WRONG PARITY
 014562  012737  000204  177746           MOV     #204,@#CCR       ;WWP OFF
 014570  005710                           TST     (R0)             ;FORCE TRAP
 014572  012737  000214  177746           MOV     #214,@#CCR       ;CACHE OFF
 014600  012737  014334  001110           MOV     #T24H10,@#$LPERR
 014606  000455                           BR      T24H15           ;REPORT ERROR

 014610  062706  000004          T24H16:  ADD     #4,SP            ;RESTORE STACK
 014614  162700  000002                   SUB     #2,R0            ;LOOK AT NEXT ADDR.

 014620  010046                           MOV     R0,-(SP)         ;SAVE R0 FOR MED INST
 014622  076600                           MED                      ;GET CONTENTS OF LOG REG
 014624  000022                           .WORD   RLOG
 014626  052700  100001                   BIS     #100001,R0       ;ENABLE ERROR LOG & LOG FIRST MODE
 014632  076600                           MED                      ;UNLOCK ERROR LOG
 014634  000222                           .WORD   WLOG
 014636  012600                           MOV     (SP)+,R0         ;RESTORE R0

 014640  020027  061776                   CMP     R0,#BUFH-2       ;ALL ADDR TESTED?
 014644  001326                           BNE     T24H17           ;BRANCH IF NO

 014646                          T24H18:
```

```

                                          ;RID CACHE OF BAD PARITY
 014646  012737  000214  177746           MOV     #214,@#CCR       ;CACHE OFF IF ON
 014654  004737  035134                   JSR     PC,SWEEP         ;GO PURGE CACHE


 014660  012737  033142  000114           MOV     #UPERR,@#PVEC
 014666  000444                           BR      TST21            ;;GO TO NEXT TEST

 014670  012737  000214  177746  T24H01:  MOV     #214,@#CCR       ;CACHE OFF
 014676  005037  001160                   CLR     $REG1            ;SAVE FAILING ADDR
 014702  010037  001162                   MOV     R0,$REG2         ;SAVE FAILING ADDR
 014706  104043                           ERROR   43               ;ERROR: ADDRESS COULD NOT BE MADE A HIT
 014710  000756                           BR      T24H18           ;GO TO END OF TEST

 014712  012737  000214  177746  T24H03:  MOV     #214,@#CCR       ;CACHE OFF
 014720  005037  001160          T24H14:  CLR     $REG1            ;SAVE FAILING ADDRESS
 014724  010037  001162                   MOV     R0,$REG2         ;SAVE FAILING ADDRESS
 014730  104111                           ERROR   111              ;ERROR: TEST OF VALID BIT FAILED
                                                                   ;        LOC COULD NOT BE MADE A HIT
 014732  000745                           BR      T24H18           ;GO TO END OF TEST

 014734  012737  000214  177746  T24H04:  MOV     #214,@#CCR       ;CACHE OFF
 014742  005037  001160          T24H15:  CLR     $REG1            ;SAVE FAILING ADDRESS
 014746  010037  001162                   MOV     R0,$REG2         ;SAVE FAILING ADDRESS
 014752  104042                           ERROR   42               ;ERROR: NO PARITY TRAP FROM LOC WRITTEN WITH WRONG PARIT
 014754  000734                           BR      T24H18           ;GO TO END OF TEST

 014756  012737  000214  177746  T24H06:  MOV     #214,@#CCR       ;CACHE OFF
 014764  005037  001160                   CLR     $REG1            ;SAVE FAILING ADDR
 014770  010037  001162                   MOV     R0,$REG2         ;SAVE FAILING ADDR
 014774  104112                           ERROR   112              ;ERROR: TEST OF VALID BIT FAILED
                                                                   ;        LOCATION NOT INVALIDATED BY PARITY TRAP
 014776  000723                           BR      T24H18           ;GO TO END OF TEST

                                 ;;****************************************************************
                                 ;*TEST 21        TEST TAG PARITY BIT FOR HIGH CACHE ADDRESSES
                                 ;*
                                 ;*
                                 ;*   THE TEST OF THE TAG PARITY BIT IS NOT COMPLETE UNTIL
                                 ;*THE TAG P BIT TEST FOR THE SECOND HALF OF CACHE AND THE
                                 ;*MSB ADDRESS (A10) TO CACHE TAG FIELD TEST ARE RUN.  TWO
                                 ;*TAG ADDRESSES ARE USED TO GENERATE A PARITY BIT OF 1 AND
                                 ;*0.  THE FIRST ADDRESS IS CHOSEN FROM A TEST BUFFER AREA
                                 ;*AND THE SECOND IS CHOSEN TO LIE 1K AWAY.  A WRITE/READ
                                 ;*PROCEDURE IS DONE WHICH CHECKS THE P BIT AND DUAL ADD-
                                 ;*RESSING FOR HALF OF CACHE.  INITIALLY THE P BIT IS WRITTEN
                                 ;*WITH ONE PARITY PATTERN IN HALF OF CACHE.  THEN STARTING
                                 ;*AT THE LOW HALF CACHE ADDRESS, THE LOC IS READ AND THEN
                                 ;*WRITTEN WITH THE OPPOSITE PARITY.  THIS IS SEQUENTIALLY
                                 ;*REPEATED WITH INCREASING ADDRESSES UNTIL THE HIGH HALF
                                 ;*CACHE ADDRESS IS REACHED.  THEN STARTING AT THE HIGH ADDR,
                                 ;*THE SECOND PARITY PATTERN IS READ AND THE LOC IS REWRITTEN
                                 ;*WITH THE FIRST.  THIS IS SEQUENTIALLY REPEATED, DECREASING
                                 ;*THE ADDRESS, UNTIL THE LOW HALF CACHE ADDRESS IS REACHED.
                                 ;*A SECOND PASS IS THEN MADE WITH THE PARITY PATTERN RE-
```

```
                                            ;*VERSED.  A PARITY ERROR HANDLER IS SETUP TO DETECT PARITY
                                            ;*ERRORS.  ALSO, LOCS WHICH SHOULD BE HITS ARE CHECKED FOR
                                            ;*AND REPORTED IF NO HIT OCCURRED.
                                            ;*
                                            ;*R0, R1 CONTAIN ADDRESSES TO GENERATE COMPLIMENTARY TAG
                                            ;*PARITY BITS.

                                            ;;*****************************************************************
015000  012737  000214  177746  TST21:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
015006  000004                          SCOPE
015010  012737  016000  001234          MOV     #TST22,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
015016  012737  015172  000114          MOV     #T11H01,@#PVEC  ;SET UP FOR PARITY ERRORS
015024  005003                          CLR     R3              ;INIT FLAG=FIRST PASS
015026  012700  062000                  MOV     #BUFH,R0        ;SET UP ADDR. FOR FIRST PASS
015032  012737  000204  177746          MOV     #204,@#CCR      ;TURN HALF CACHE ON
015040  012701  001000          T11H02: MOV     #1000,R1        ;INIT COUNTER
015044  005720                  1$:     TST     (R0)+           ;PUT PARITY PATTERN IN TAG FIELD
015046  077102                          SOB     R1,1$           ;LOAD HALF OF CACHE

015050  012701  001000                  MOV     #1000,R1        ;INIT. COUNTER
015054  012700  062000                  MOV     #BUFH,R0        ;SET UP ADDR. FOR FIRST PASS
015060  012702  056000                  MOV     #BUFH-4000,R2   ;SET UP ADDR. FOR FIRST PASS
015064  005703                          TST     R3              ;FIRST PASS?
015066  001404                          BEQ     T11H03          ;BRANCH IF YES
015070  012700  056000                  MOV     #BUFH-4000,R0   ;SET UP ADDR. FOR SECOND PASS
015074  012702  062000                  MOV     #BUFH,R2        ;SET UP ADDR. FOR SECOND PASS
015100  005720                  T11H03: TST     (R0)+           ;READ CACHE TO SEE IF PARITY OK; NO-TRAPS
015102  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS ADDRESS A HIT?
015110  001533                          BEQ     T11H04          ;BRANCH TO ERROR IF NO
015112  005722                          TST     (R2)+           ;WRITE DIFFERENT PARITY PATTERN IN TAG FIELD
015114  077107                          SOB     R1,T11H03       ;LOOK AT HALF OF CACHE

015116  012701  001000                  MOV     #1000,R1        ;INIT COUNTER
015122  005742                  T11H11: TST     -(R2)           ;READ SECOND PARITY PATTERN
015124  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS ADDRESS A HIT?
015132  001532                          BEQ     T11H05          ;BRANCH IF NO TO ERROR
015134  005740                          TST     -(R0)           ;PUT NEW PARITY PATTERN IN TAG
015136  077107                          SOB     R1,T11H11       ;LOOK AT HALF OF CACHE

015140  005703                          TST     R3              ;FIRST PASS?
015142  001140                          BNE     T11H06          ;NO GO TO END OF TEST
015144  052703  000001                  BIS     #1,R3           ;SET FLAG TO INDIC. SECOND PASS
015150  012737  000204  177746  T11H12: MOV     #204,@#CCR      ;HALF CACHE ON IF OFF
015156  012737  015150  001110          MOV     #T11H12,@#$LPERR  ;SETUP RETURN FOR ERROR IF ONE OCCURS
015164  012700  056000                  MOV     #BUFH-4000,R0   ;SET UP FOR SECOND PASS.
015170  000723                          BR      T11H02          ;GO TEST SECOND PASS

015172                          T11H01:

                                        ;RID CACHE OF BAD PARITY
015172  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
015200  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE



015204  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
```

```
015206  076600                          MED                     ;GET CONTENTS OF LOG REG
015210  000022                          .WORD   RLOG
015212  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
015216  076600                          MED                     ;UNLOCK ERROR LOG
015220  000222                          .WORD   WLOG
015222  012600                          MOV     (SP)+,R0        ;RESTORE R0

015224  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
015226  000101                          .WORD   RSER
015230  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
015232  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
015236  010037  001160                  MOV     R0,@REG1        ;SAVE ADDRESS
015242  076600                          MED                     ;GET LOG INFORMATION
015244  000102                          .WORD   LOADD
015246  010037  001162                  MOV     R0,@REG2        ;SAVE INFORMATION
015252  076600                          MED                     ;GET LOG INFORMATION
015254  000100                          .WORD   RJAM
015256  032700  000400                  BIT     #400,R0         ;ERROR IN BACKING STORE?
015262  001410                          BEQ     T11H07          ;BRANCH IF NO
015264  011637  001164                  MOV     (SP),@REG3      ;GET PC+2 WHERE ERROR OCCURRED
015270  162737  000002  001166          SUB     #2,@REG4        ;SAVE PC WHERE ERROR OCCURRED
015276  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
015300  104001                          ERROR   1               ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE
015302  000460                          BR      T11H06          ;GO TO NEXT TEST

015304  022626                  T11H07: CMP     (SP)+,(SP)+     ;RESTORE STACK
015306  032737  000040  177744          BIT     #40,@#EREG      ;ERROR IN TAG?
015314  001411                          BEQ     T11H08          ;BRANCH NO
015316  076600                          MED                     ;GET TAG LOG INFO.
015320  000107                          .WORD   RTAG
015322  000300                          SWAB    R0              ;PUT TAG IN LOW BYTE
015324  042700  177400                  BIC     #177400,R0      ;LOOK AT TAG ONLY
015330  010037  001164                  MOV     R0,@REG3        ;SAVE BAD DATA
015334  104045                          ERROR   45              ;ERROR: TAG PARITY ERROR WHEN TESTING TAG P BIT
015336  000442                          BR      T11H06          ;GO TO NEXT TEST

015340  032737  000100  177744  T11H08: BIT     #100,@#EREG     ;ERROR IN LOW BYTE?
015346  001406                          BEQ     T11H09          ;BRANCH IF NO
015350  076600                          MED                     ;GET LOG INFORMATION
015352  000106                          .WORD   CDL
015354  010037  001164                  MOV     R0,@REG3        ;SAVE INFORMATION
015360  104046                          ERROR   46              ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING TAG P BIT
015362  000430                          BR      T11H06          ;NEXT TEST

015364                          T11H09:
015364  076600                          MED                     ;GET LOG INFORMATION
015366  000106                          .WORD   CDH
015370  010037  001164                  MOV     R0,@REG3        ;SAVE INFORMATION
015374  104047                          ERROR   47              ;ERROR: HIGH BYTE PARITY ERROR WHEN TESTING TAG P BIT
015376  000422                          BR      T11H06          ;NEXT TEST

015400  052737  000014  177746  T11H04: BIS     #14,@#CCR       ;CACHE OFF
015406  162700  000002                  SUB     #2,R0           ;GET BAD ADDRESS
015412  010037  001162                  MOV     R0,@REG2        ;SAVE BAD ADDRESS
015416  000407                          BR      T11H10          ;REPORT ERROR
015420  052737  000014  177746  T11H05: BIS     #14,@#CCR       ;CACHE OFF
```

```
015426  010237  001162                    MOV     R2,$REG2        ;SAVE BAD ADDRESS
015432  062702  000002                    ADD     #2,R2           ;RESTORE R2 TO FAILING ADDR.+2
015436  005037  001160            T11H10: CLR     $REG1           ;SAVE BAD ADDRESS
015442  104043                            ERROR   43              ;ERROR; ADDRESS COULD NOT BE MADE A HIT

015444  012737  033142  000114    T11H06: MOV     #UPERR,@#PVEC   ;RESTORE PARITY TRAP HANDLER
015452  052737  000014  177746            BIS     #14,@#CCR       ;CACHE OFF WHEN CROSS CACHE ADDR. BOUNDARY
015460  000547                            BR      TST22           ;;GO TO NEXT TEST


        016000                            .=16000                 ;ADJUST ADDRESS SPACE FOR NEXT TEST

                                  ;;*****************************************************************
                                  ;*TEST 22        TEST TAG ADDRESS BITS FOR LOW HALF OF CACHE
                                  ;*
                                  ;*   THE TEST OF THE TAG BITS IS NOT COMPLETE UNTIL THE
                                  ;*TAG ADDRESS TEST FOR THE OTHER HALF OF CACHE AND THE
                                  ;*TEST OF THE MSB ADDRESS (A10) TO THE CACHE TAG FIELD
                                  ;*ARE RUN.  A WRITE/READ PROCEDURE IS DONE WHICH CHECKS
                                  ;*THE TAG FIELD BITS AND DUAL ADDRESSING ON THEM FOR HALF
                                  ;*OF CACHE.  MEMORY IS FIRST SIZED TO DETERMINE THE MAX-
                                  ;*IMUM TESTABLE ADDRESS.  THE TAG ADDRESS BITS OF THIS
                                  ;*ADDRESS ARE USED AS PATTERN A AND STORED IN KIPAR4.  A
                                  ;*PATTERN B IS NOW GENERATED WHICH HAS 'COMPLEMENT' TAG
                                  ;*BITS AND STORED IN KIPAR5.  ON THE FIRST PASS, PATTERN
                                  ;*A IS WRITTEN THROUGH HALF OF CACHE.  NEXT, STARTING AT
                                  ;*THE HIGH HALF CACHE ADDRESS, THE LOCATION IS READ,
                                  ;*CHECKED TO BE A HIT AND THEN WRITTEN WITH PATTERN B.
                                  ;*THIS IS SEQUENTIALLY REPEATED WITH DECREASING ADDRESSES
                                  ;*UNTIL THE LOW HALF CACHE ADDRESS IS REACHED.  AT THE
                                  ;*LOW ADDRESS, THE SECOND PATTERN IS READ, CHECKED TO BE A
                                  ;*HIT AND REWRITTEN WITH THE FIRST PATTERN.  THIS IS SE-
                                  ;*QUENTIALLY REPEATED WITH INCREASING ADDRESSES UNTIL THE
                                  ;*HIGH HALF CACHE ADDRESS IS REACHED.  A SECOND PASS IS
                                  ;*THEN MADE WITH THE PATTERNS REVERSED.
                                  ;*   ANY PARITY ERROR OR HIT ERROR IS REPORTED.
                                  ;*   DURING THE PASSES, R0, R1 CONTAIN ADDRESSES WHICH
                                  ;*REFERENCE KIPAR5,5.
                                  ;*   R3 INDICATES THE PASS NUMBER.
                                  ;*IF THE INHIBIT TESTS USING KT SWITCH (SW12) IS SET, THIS
                                  ;*TEST IS SKIPPED.

                                  ;;*****************************************************************
016000  012737  000214  177746    TST22:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
016006  000004                            SCOPE
016010  012737  016646  001234            MOV     #TST23,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
016016  032777  010000  163110            BIT     #SW12,@SWR      ;INHIBIT TESTS USING KT?
016024  001402                            BEQ     3$              ;CONTINUE TEST IF NO
016026  000137  016646                    JMP     @#TST23         ;GO TO NEXT TEST
016032  052737  000200  036034    3$:     BIS     #200,@#$KT11    ;KT ON FOR $SIZE
016040  004737  035750                    JSR     PC,$SIZE        ;SIZE MEMORY
016044  012737  016266  000114            MOV     #T13L01,@#PVEC  ;SET UP PARITY ERROR HANDLER
016052  013737  036322  172350            MOV     @#$LSTBK,@#KIPAR4  ;SET UP PAR4 FOR ADDRESS PATTERN A

                                  ;CALC COMPLEMENT TAG PATTERN B
```

```
016060  013700  036322                    MOV     @#$LSTBK,R0     ;GET TEST PATTERN A AND
016064  005100                            COM     R0              ;CALC PATTERN B
016066  005001                            CLR     R1
016070  005201                    1$:     INC     R1
016072  006300                            ASL     R0
016074  100775                            BMI     1$
016076  006200                    2$:     ASR     R0
016100  077102                            SOB     R1,2$
016102  042700  000037                    BIC     #37,R0          ;ONLY COMPLEMENT TAG ADDR. BITS

016106  010037  172352                    MOV     R0,@#KIPAR5     ;SET UP PAR5 FOR ADDRESS PATTERN B

016112  012700  100000                    MOV     #100000,R0      ;INIT R0 TO ADD PATTERN A
016116  012701  122000                    MOV     #122000,R1      ;INIT R1 TO ADD PATTERN B
016122  005003                            CLR     R3              ;INIT FLAG FOR PASS 1
016124  005004                    T13L02: CLR     R4              ;INIT INDICATOR FOR ERROR LOOP 1

016126  012702  001000                    MOV     #1000,R2        ;INIT ADDR. COUNTER
016132  052737  000001  177572            BIS     #1,@#MMR0       ;TURN KT ON
016140  012737  000210  177746            MOV     #210,@#CCR      ;TURN HALF OF CACHE ON

016146  005720                    1$:     TST     (R0)+           ;WRITE PATTERN IN CACHE
016150  077202                            SOB     R2,1$           ;ALL DONE? BRANCH IF NO

016152  012702  001000                    MOV     #1000,R2        ;INIT. ADDR. COUNTER
016156  005740                    T13L03: TST     -(R0)           ;READ CACHE TAG BITS
016160  033727  177752  000004            BIT     @#HMR,#HMR2     ;HIT?
016166  001002                            BNE     2$              ;BRANCH  IF YES
016170  000137  016540                    JMP     T13L04          ;REPORT ERROR
016174  005741                    2$:     TST     -(R1)           ;WRITE NEW PATTERN IN TAG
016176  077211                            SOB     R2,T13L03               ;HALF ADDR. TESTED? BRANCH IF NO

016200  005204                            INC     R4              ;SET INDICATOR FOR ERROR LOOP 2
016202  012702  001000                    MOV     #1000,R2        ;INIT. ADDR. COUNTER
016206  005711                    T13L05: TST     (R1)            ;READ CACHE TAG BITS
016210  033727  177752  000004            BIT     @#HMR,#HMR2     ;HIT?
016216  001002                            BNE     3$              ;BRANCH IF YES
016220  000137  016606                    JMP     T13L06          ;REPORT ERROR
016224  005721                    3$:     TST     (R1)+           ;UPDATE FOR NEXT ADDRESS
016226  005720                            TST     (R0)+           ;WRITE NEW PATTERN IN TAG
016230  077212                            SOB     R2,T13L05

016232  005703                            TST     R3              ;SECOND PASS?
016234  001402                            BEQ     2$              ;CONTINUE TEST IF NO
016236  000137  016634                    JMP     T13L07          ;GO TO END OF TEST
016242  052703  000001            2$:     BIS     #1,R3           ;SET FLAG FOR SECOND PASS
016246  012737  016254  001110            MOV     #T13L15,@#$LPERR  ;INIT RETURN FOR ERROR LOOP IF ERROR OCCURS
016254  012700  120000            T13L15: MOV     #120000,R0      ;INIT. R0 TO ADDR. PATTERN B
016260  012701  102000                    MOV     #102000,R1      ;INIT. R1 TO ADDR. PATTERN A
016264  000717                            BR      T13L02          ;GO TEST SECOND PASS

016266  052737  000014  177746    T13L01: BIS     #14,@#CCR       ;CACHE OFF

016274  010046                            MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
016276  076600                            MED                     ;GET CONTENTS OF LOG REG
016300  000022                            .WORD   RLOC
016302  052700  100001                    BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
```

```
016306  076600                           MED                         ;UNLOCK ERROR LOG
016310  000222                           .WORD   WLOG
016312  012600                           MOV     (SP)+,R0            ;RESTORE R0

016314  011637  001164                   MOV     (SP),$REG3          ;GET PC+2 OF TRAP
016320  162737  000002  001164           SUB     #2,$REG3            ;SAVE PC FOR MAIN PARITY ERROR
016326  022626                           CMP     (SP)+,(SP)+         ;RESTORE STACK
016330  010046                           MOV     R0,-(SP)            ;SAVE R0 ON STACK FOR MED INST.
016332  076600                           MED                         ;GET LOG INFOR FOR PHY. ADDR. A17,A16
016334  000101                           .WORD   RSER
016336  000300                           SWAB    R0                  ;PUT PHY. ADDR A17, A16 IN LOW BYTE
016340  042700  177776                   BIC     #177776,R0          ;ONLY LOOK AT A17, A16
016344  010037  001160                   MOV     R0,$REG1            ;SAVE ADDRESS
016350  076600                           MED                         ;GET LOG INFORMATION
016352  000102                           .WORD   LOADD
016354  010037  001162                   MOV     R0,$REG2            ;SAVE INFORMATION
016360  076600                           MED                         ;GET LOG INFORMATION
016362  000100                           .WORD   RJAM
016364  012600                           MOV     (SP)+,R0            ;RESTORE R0
016366  032700  000400                   BIT     #400,R0             ;ERROR BACKING STORE?
016372  001402                           BEQ     T13L08              ;BRANCH IF NO
016374  104001                           ERROR   1                   ;ERROR; UNEXPECT. PARITY ERROR IN BACKING STORE
016376  000516                           BR      T13L07              ;GO TO END OF TEST

016400  011137  001166           T13L08: MOV     (R1),$REG4          ;SAVE GOOD DATA
016404  005704                           TST     R4                  ;ERROR IN LOOP 2?
016406  001002                           BNE     T13L09              ;BRANCH IF YES
016410  011037  001166                   MOV     (R0),$REG4          ;SAVE GOOD DATA

016414  032737  000040  177744   T13L09: BIT     #40,@#EREG          ;TAG PARITY ERROR?
016422  001426                           BEQ     T13L10              ;BRANCH IF NO
016424  004737  033634                   JSR     PC,PAR              ;GET PAR USED
016430  000000                           .WORD   0                   ;INDICATOR FOR R0
016432  005704                           TST     R4                  ;ERROR FROM LOOP 1?
016434  001403                           BEQ     T13L11              ;BRANCH IF YES
016436  004737  033634                   JSR     PC,PAR              ;GET PAR USED
016442  000001                           .WORD   1                   ;INDICATOR FOR R1
016444  004737  033606           T13L11: JSR     PC,TAG              ;CALC TAG CONTENTS
016450  013737  001172  001166           MOV     $TMP0,$REG4         ;SAVE GOOD DATA
016456  076600                           MED                         ;GET TAG LOG INFO.
016460  000107                           .WORD   RTAG
016462  000300                           SWAB    R0                  ;PUT TAG IN LOW BYTE
016464  042700  177400                   BIC     #177400,R0          ;LOOK AT TAG ONLY
016470  010037  001164                   MOV     R0,$REG3            ;SAVE BAD DATA
016474  104052                           ERROR   52                  ;ERROR: TAG PARITY ERROR ON TEST OF TAG ADDRESS BITS
016476  000456                           BR      T13L07              ;GO TO END OF TEST

016500  032737  000100  177744   T13L10: BIT     #100,@#EREG         ;LOW BYTE P.E.?
016506  001406                           BEQ     T13L12              ;BRANCH IF NO
016510  076600                           MED                         ;GET LOG INFORMATION
016512  000106                           .WORD   CDL
016514  010037  001164                   MOV     R0,$REG3            ;SAVE INFORMATION
016520  104053                           ERROR   53                  ;ERROR: LOW BYTE PARITY ERROR ON TEST OF TAG ADDR. BITS
016522  000444                           BR      T13L07              ;GO TO END OF TEST

016524                           T13L12:
```

```
016524  076600                           MED                         ;GET LOG INFORMATION
016526  000106                           .WORD   CDH
016530  010037  001164                   MOV     R0,$REG3            ;SAVE INFORMATION
016534  104054                           ERROR   54                  ;ERROR; HIGH BYTE PARITY ERROR ON TEST OF TAG ADDR. BITS
016536  000436                           BR      T13L07              ;GO TO END OF TEST

016540  052737  000014  177746   T13L04: BIS     #14,@#CCR           ;CACHE OFF
016546  010037  001172                   MOV     R0,$TMP0            ;GET VIRTUAL ADDRESS TESTED
016552  004737  033434                   JSR     PC,VIP              ;SAVE ADDRESS TESTED
016556  062700  000002                   ADD     #2,R0               ;ADJUST ADDRESS WHEN LOOP
016562  004737  033634                   JSR     PC,PAR              ;GET PAR TESTED
016566  000000                           .WORD   0                   ;INDICATOR FOR R0
016570  004737  033606           T13L13: JSR     PC,TAG              ;CALC TAG FROM PAR
016574  013737  001172  001164           MOV     $TMP0,$REG3         ;SAVE TAG
016602  104055                           ERROR   55                  ;ERROR: TEST OF TAG ADDRESS BITS FAILED
                                                                     ;        ADDR. COULD NOT BE MADE A HIT
016604  000413                           BR      T13L07              ;GO TO NEXT TEST

016606  052737  000014  177746   T13L06: BIS     #14,@#CCR           ;CACHE OFF
016614  010137  001172                   MOV     R1,$TMP0            ;GET VIRTUAL ADDRESS TESTED
016620  004737  033434                   JSR     PC,VIP              ;SAVE PHYSICAL ADDRESS TESTED
016624  004737  033634                   JSR     PC,PAR              ;GET PAR TESTED
016630  000001                           .WORD   1                   ;INDICATOR FOR R1
016632  000756                           BR      T13L13              ;REPORT ERROR


016634  005037  177572           T13L07: CLR     @#MMR0              ;KT OFF
016640  012737  033142  000114           MOV     #UPERR,@#PVEC       ;RESTORE UNEXP. PARITY ERROR HANDLER

                                 ;;*****************************************************************
                                 ;*TEST 23       TEST OF CACHE DATA LOC WITH FLOAT 1 & 0 PATTERNS
                                 ;*
                                 ;*   THIS TEST MAKES TWO PASSES.  ON THE FIRST, A FLOAT
                                 ;*'1' PATTERN IS WRITTEN/READ FROM ONE CACHE LOC.  ON THE
                                 ;*SECOND, A FLOAT '0' PATTERN IS WRITTEN/READ FROM ONE
                                 ;*CACHE LOC.  THERE IS A HANDLER FOR PARITY ERRORS.  IF
                                 ;*THERE ARE LESS THAN 4 PARITY ERRORS THE TEST CONTINUES.
                                 ;*IF THERE ARE 4 OR MORE PARITY ERRORS THE TEST IS STOPPED.
                                 ;*   R0 CONTAINS THE DATA PATTERN
                                 ;*   R2 CONTAINS THE TEST ADDRESS
                                 ;*   R4 IS THE PASS INDICATOR

                                 ;;*****************************************************************
016646  012737  000214  177746   TST23:  MOV     #214,@#CCR          ;CACHE OFF FOR SCOPE
016654  000004                           SCOPE
016656  012737  020000  001234           MOV     #TST24,SKTST        ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
016664  012737  016760  000114           MOV     #T14LP1,@#PVEC      ;SET UP PARITY ERROR HANDLER
016672  005004                           CLR     R4                  ;CLEAR PASS INDICATOR FOR FIRST PASS
016674  012700  000001                   MOV     #1,R0               ;SET UP FLOAT 1 PATTERN
016700  012702  060000                   MOV     #BUFL,R2            ;SET UP TEST ADDRESS
016704  012737  000210  177746   T14L02: MOV     #210,@#CCR          ;HALF CACHE ON
016712  010012                   T14L06: MOV     R0,(R2)             ;WRITE CACHE
016714  020012                           CMP     R0,(R2)             ;READ CACHE
016716  001151                           BNE     T14L03              ;BRANCH TO ERROR IF DATA BAD
016720  005704                   T14L10: TST     R4                  ;FIRST PASS?
016722  001011                           BNE     T14L04              ;BRANCH IF NO
```

```
016724  005700                          TST     R0              ;ALL SHIFTS FOR FLOAT 1 PATTERN DONE?
016726  100402                          BMI     T14L05          ;BRANCH IF YES
016730  006300                          ASL     R0              ;SHIFT FLOAT 1 PATTERN
016732  000767                          BR      T14L06          ;TEST IT

016734  052704  000001          T14L05: BIS     #1,R4           ;SET FLAG FOR SECOND PASS
016740  012700  177776                  MOV     #177776,R0      ;SET UP FLOAT 0 PATTERN
016744  000762                          BR      T14L06          ;GO TEST IT

016746  005700                  T14L04: TST     R0              ;ALL SHIFTS FOR FLOAT 0 PATTERN DONE?
016750  100155                          BPL     T14L07          ;GO TO END OF TEST IF YES
016752  000261                          SEC                     ;SET CARRY BIT FOR ROTATE
016754  006100                          ROL     R0              ;ROTATE FLOAT 0 PATTERN
016756  000755                          BR      T14L06          ;TEST IT

016760  052737  000014  177746  T14L01: BIS     #14,@#CCR       ;CACHE OFF

016766  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
016770  076600                          MED                     ;GET CONTENTS OF LOG REG
016772  000022                          .WORD   RLOG
016774  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
017000  076600                          MED                     ;UNLOCK ERROR LOG
017002  000222                          .WORD   WLOG
017004  012600                          MOV     (SP)+,R0        ;RESTORE R0

017006  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 OF ERROR
017012  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF ERROR
017020  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
017022  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
017024  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR, A17,A16
017026  000101                          .WORD   RSER
017030  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
017032  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
017036  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
017042  076600                          MED                     ;GET LOG INFORMATION
017044  000102                          .WORD   LOADD
017046  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
017052  076600                          MED                     ;GET LOG INFORMATION
017054  000100                          .WORD   RJAM
017056  032700  000400                  BIT     #400,R0         ;ERROR IN BACKING STORE?
017062  001403                          BEQ     T14L08          ;BRANCH IF NO
017064  010026                          MOV     R0,(SP)+        ;RESTORE R0
017066  104001                          ERROR   1               ;ERROR: UNEXPECT. PARITY ERROR IN BACKING STORE
017070  000505                          BR      T14L07          ;GO TO END OF TEST

017072  011637  001166          T14L08: MOV     (SP),$REG4      ;SAVE GOOD DATA
017076  012737  016704  001110          MOV     #T14L02,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
017104  032737  000100  177744          BIT     #100,@#EREG     ;LOW BYTE PARITY ERROR?
017112  001416                          BEQ     T14L09          ;BRANCH IF NO
017114  076600                          MED                     ;GET LOG INFORMATION
017116  000106                          .WORD   CDL
017120  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
017124  012600                          MOV     (SP)+,R0        ;RESTORE R0
017126  104056                          ERROR   56              ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA FIELD
017130  123727  001103  000003  T14L12: CMPB    @#$ERFLG,#3     ;MORE THAN 3 ERRORS?
017136  101062                          BHI     T14L07          ;STOP TESTING IF YES
```

```
017140  012737  000210  177746          MOV     #210,@#CCR      ;HALF CACHE ON
017146  000664                          BR      T14L10          ;CONTINUE TEST

017150  032737  000200  177744  T14L09: BIT     200,@#EREG      ;HIGH BYTE P.E.?
017156  001407                          BEQ     T14L11          ;BRANCH IF NO
017160  076600                          MED                     ;GET LOG INFORMATION
017162  000106                          .WORD   CDH
017164  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
017170  012600                          MOV     (SP)+,R0        ;RESTORE R0
017172  104057                          ERROR   57              ;ERROR: HIGH BYTE PARITY ERROR WHEN TESTING DATA FIELD
017174  000755                          BR      T14L12          ;SEE IF SHOULD CONTINUE TESTING

017176                          T14L11:
017176  076600                          MED                     ;GET LOG INFORMATION
017200  000107                          .WORD   CTAG
017202  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
017206  012600                          MOV     (SP)+,R0        ;RESTORE R0
017210  012737  060000  001166          MOV     #BUFL,$REG4     ;GET TESTED ADDRESS
017216  012705  000013                  MOV     #13,R5          ;SETUP COUNTER
017222  006237  001166          2$:     ASR     $REG4           ;PUT TAG ADDRESS BITS IN LSB 6-0
017226  077503                          SOB     R5,2$           ;SHIFT NINE PLACES
017230  052737  000200  001166          BIS     #200,$REG4      ;SET VALID BIT
017236  104060                          ERROR   60              ;ERROR: TAG PARITY ERROR WHEN TESTING CACHE DATA FIELD
017240  000733                          BR      T14L12          ;SEE IF WANT TO CONTINUE TEST

017242  011205                  T14L03: MOV     (R2),R5         ;GET BAD DATA
017244  052737  000014  177746          BIS     #14,@#CCR       ;CACHE OFF
017252  005037  001160                  CLR     $REG1           ;SAVE ADDRESS
017256  010237  001162                  MOV     R2,$REG2        ;SAVE ADDRESS
017262  010537  001164                  MOV     R5,$REG3        ;SAVE BAD DATA
017266  010037  001166                  MOV     R0,$REG4        ;SAVE GOOD DATA
017272  012737  016704  001110          MOV     #T14L02,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
017300  104061                          ERROR   61              ;ERROR: CACHE DATA LOC HELD WRONG DATA
017302  000712                          BR      T14L12          ;SEE IF TEST TO BE CONTINUED

017304  012737  033142  000114  T14L07: MOV     #UPERR,@#PVEC   ;RESTORE HANDLER FOR UNEXP. PARITY ERRORS
017312  052737  000014  177746          BIS     #14,@#CCR       ;CACHE OFF WHEN CROSS CACHE ADDR. BOUNDARY
017320  000137  020000                  JMP     @#TST24         ;GO TO NEXT TEST


        020000                          .=20000                 ;ADJUST ADDRESS SPACE FOR NEXT TEST


                                ;;*************************************************************
                                ;*TEST 24        TEST DATA PARITY BITS FOR HIGH CACHE
                                ;*
                                ;*   THE TEST OF THE DATA PARITY BITS ARE NOT COMPLETE
                                ;*UNTIL THE DATA P BIT TEST FOR THE SECOND HALF OF CACHE
                                ;*AND THE MSB ADDRESS (A10) TO CACHE DATA FIELD ARE RUN.
                                ;*A WRITE/READ PROCEDURE IS DONE WHICH SIMULTANEOUSLY
                                ;*CHECKS THE DATA P BIT FOR BOTH BYTES AND DUAL ADDRESSING
                                ;*IN HALF OF CACHE FOR IT.  INITIALLY THE P BIT IS WRITTEN
                                ;*WITH ONE PARITY PATTERN IN HALF OF CACHE.  THEN STARTING
                                ;*AT THE LOW HALF CACHE ADDRESS, THE LOC IS READ AND THEN
                                ;*WRITTEN WITH THE OPPOSITE PARITY.  THIS IS  SEQUEN-
                                ;*TIALLY REPEATED WITH INCREASING ADDRESSES UNTIL THE HIGH
```

```
                                                   ;*HALF CACHE ADDRESS IS REACHED.  THEN STARTING AT THE
                                                   ;*HIGH ADDR, THE SECOND PARITY PATTERN IS READ AND THE LOC
                                                   ;*IS REWRITTEN WITH THE FIRST.  THIS IS SEQUENTIALLY RE-
                                                   ;*PEATED DECREASING THE ADDRESS UNTIL THE LOW HALF CACHE
                                                   ;*ADDRESS IS REACHED.  A SECOND PASS IS THEN MADE WITH
                                                   ;*THE PARITY PATTERN REVERSED.  A PARITY ERROR HANDLER IS
                                                   ;*SETUP TO DETECT PARITY ERRORS.  ALSO, LOCS WHICH SHOULD
                                                   ;*BE HITS ARE CHECKED FOR AND REPORTED IF NO HIT OCCURRED.
                                                   ;*
                                                   ;*R0, R1 CONTAIN DATA WHICH GENERATE OPPOSITE PARITY.  R3
                                                   ;*INDICATES WHICH PASS IS BEING DONE.

                                                   ;;*****************************************************************
      020000  012737  000214  177746   TST24:  MOV     #214,@#CCR          ;CACHE OFF FOR SCOPE
      020006  000004                           SCOPE
      020010  012737  020456  001234           MOV     #TST25,SKTST        ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR -
      020016  012737  020200  000114           MOV     #T12H01,@#PVEC      ;SET UP PARITY ERROR HANDLER
      020024  005003                           CLR     R3                  ;INIT FLAG FOR FIRST PASS
      020026  005000                           CLR     R0                  ;SET UP PARITY PATTERN A FOR FIRST PASS
      020030  012737  000204  177746   T12H02: MOV     #204,@#CCR          ;HALF CACHE ON
      020036  012701  001000                   MOV     #1000,R1            ;INIT ADDR. COUNTER
      020042  012705  062000                   MOV     #BUFH,R5            ;INIT. TEST ADDRESS
      020046  010025                   1$:     MOV     R0,(R5)+            ;WRITE DATA PARITY PATTERN
      020050  077102                           SOB     R1,1$               ;HALF ADDR. WRITTEN? BRANCH IF NO

      020052  012701  001000                   MOV     #1000,R1            ;INIT ADDR. COUNTER
      020056  012705  062000                   MOV     #BUFH,R5            ;INIT. TEST ADDR
      020062  012700  000401                   MOV     #401,R0             ;SET UP PATTERN B FOR FIRST PASS
      020066  005703                           TST     R3                  ;FIRST PASS?
      020070  001401                           BEQ     2$                  ;BRANCH IF YES
      020072  005000                           CLR     R0                  ;SET UP PARITY PATTERN A FOR SECOND PASS
      020074  005715                   2$:     TST     (R5)                ;SEE IF PARITY UNCHANGED
      020076  033727  177752  000004           BIT     @#HMR,#HMR2         ;DATA FROM CACHE?
      020104  001551                           BEQ     T12H07              ;BRANCH TO ERROR IF NO
      020106  010025                           MOV     R0,(R5)+            ;WRITE NEW DATA PARITY PATTERN
      020110  077107                           SOB     R1,2$               ;HALF ADDR. SPACE EXAMINED & WRITTEN?

      020112  012701  001000                   MOV     #1000,R1            ;INIT ADDR. COUNTER
      020116  005000                           CLR     R0                  ;SET UP PARITY PATTERN A FOR FIRST PASS
      020120  005703                           TST     R3                  ;FIRST PASS?
      020122  001402                           BEQ     T12H06              ;BRANCH IF YES
      020124  012700  000401                   MOV     #401,R0             ;SET UP PARITY PATTERN B FOR SECOND PASS
      020130  012737  000204  177746   T12H06: MOV     #204,@#CCR          ;HALF CACHE ON IF OFF FROM ERROR
      020136  005745                   1$:     TST     -(R5)               ;SEE IF PARITY UNCHANGED
      020140  033727  177752  000004           BIT     @#HMR,#HMR2         ;DATA FROM CACHE
      020146  001530                           BEQ     T12H07              ;BRANCH IF NO TO ERROR
      020150  010015                           MOV     R0,(R5)             ;WRITE NEW PARITY PATTERN IN CACHE
      020152  077107                           SOB     R1,1$               ;HALF OF ADDRESS SPACE READ & WRITTEN? BRANCH IF NO

      020154  005703                           TST     R3                  ;SECOND PASS?
      020156  001134                           BNE     T12H08              ;GO TO END OF TEST IF YES
      020160  012700  000401           T12H13: MOV     #401,R0             ;SET UP PARITY PATTERN B FOR SECOND PASS
      020164  052703  000001                   BIS     #1,R3               ;SET FLAG FOR PASS 2
      020170  012737  020160  001110           MOV     #T12H13,@#LPERR  ;INIT RETURN FOR ERROR LOOP IF ERROR OCCURS
      020176  000714                           BR      T12H02              ;TEST DATA

```

```
      020200  052737  000014  177746   T12H01: BIS     #14,@#CCR           ;CACHE OFF

      020206  010046                           MOV     R0,-(SP)            ;SAVE R0 FOR MED INST
      020210  076600                           MED                         ;GET CONTENTS OF LOG REG
      020212  000022                           .WORD   RLOG
      020214  052700  100001                   BIS     #100001,R0          ;ENABLE ERROR LOG & LOG FIRST MODE
      020220  076600                           MED                         ;UNLOCK ERROR LOG
      020222  000222                           .WORD   WLOG
      020224  012600                           MOV     (SP)+,R0            ;RESTORE R0

      020226  076630                           MED                         ;GET LOG INFOR FOR PHY. ADDR. A17,A16
      020230  000101                           .WORD   RSER
      020232  000300                           SWAB    R0                  ;PUT PHY. ADDR A17, A16 IN LOW BYTE
      020234  042700  177776                   BIC     #177776,R0          ;ONLY LOOK AT A17, A16
      020240  010037  001160                   MOV     R0,$REG1            ;SAVE ADDRESS
      020244  076600                           MED                         ;GET LOG INFORMATION
      020246  000102                           .WORD   LOADU
      020250  010037  001162                   MOV     R0,$REG2            ;SAVE INFORMATION
      020254  032737  000040  177744           BIT     #40,@#EREG          ;ERROR IN TAG?
      020262  001417                           BEQ     T12H09              ;BRANCH IF NO
      020264  011637  001166                   MOV     (SP),$REG4          ;GET PC+2 OF ERROR
      020270  162737  000002  001166           SUB     #2,$REG4            ;GET PC OF ERROR
      020276  076600                           MED                         ;GET TAG LOG INFO.
      020300  000107                           .WORD   RTAG
      020302  000300                           SWAB    R0                  ;PUT TAG IN LOW BYTE
      020304  042700  177400                   BIC     #177400,R0          ;LOOK AT TAG ONLY
      020310  010037  001164                   MOV     R0,$REG3            ;SAVE BAD DATA
      020314  022626                           CMP     (SP)+,(SP)+         ;RESTORE THE STACK
      020316  104002                           ERROR   2                   ;ERROR: UNEXPECTED PARITY ERROR IN TAG FIELD
      020320  000453                           BR      T12H08              ;GO TO END OF TEST

      020322  022626                   T12H09: CMP     (SP)+,(SP)+         ;RESTORE STACK
      020324  005037  001166                   CLR     $REG4               ;SAVE GOOD DATA
      020330  005700                           TST     R0                  ;WAS TEST DATA =0?
      020332  001003                           BNE     T12H11              ;BRANCH IF NO
      020334  012737  000401  001166           MOV     #401,$REG4          ;SAVE GOOD DATA
      020342  032737  000200  177744   T12H11: BIT     #200,@#EREG         ;ERROR IN HIGH BYTE?
      020350  001406                           BEQ     T12H12              ;BRANCH IF NO
      020352  076600                           MED                         ;GET LOG INFORMATION
      020354  000106                           .WORD   CDH
      020356  010037  001164                   MOV     R0,$REG3            ;SAVE INFORMATION
      020362  104050                           ERROR   50                  ;ERROR: HIGH BYTE PARITY ERROR WHEN TESTING DATA P BITS
      020364  000431                           BR      T12H08              ;GO TO END OF TEST

      020366  032777  000100  157350   T12H12: BIT     #100,@EREG          ;ERROR IN LOW BYTE?
      020374  001406                           BEQ     T12H14              ;BRANCH IF NO
      020376  076600                           MED                         ;GET LOG INFORMATION
      020400  000106                           .WORD   CDL
      020402  010037  001164                   MOV     R0,$REG3            ;SAVE INFORMATION
      020406  104051                           ERROR   51                  ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA P BITS
      020410  000417                           BR      T12H08              ;GO TO END OF TEST

      020412  016637  177774  001164   T12H14: MOV     -4(SP),$REG3        ;GET PC+2 OF TRAP
      020420  162737  000002  001164           SUB     #2,$REG3            ;SAVE PC OF TRAP
      020426  104001                           ERROR   1                   ;ERROR: UNEXP. PARITY ERROR IN BACKING STORE

```

```
020430  052737  000014  177746  T12H07: BIS     #14,@#CCR       ;CACHE OFF
020436  010537  001162                  MOV     R5,$REG2        ;SAVE BAD ADDRESS
020442  005037  001160                  CLR     $REG1           ;SAVE BAD ADDRESS
020446  104043                          ERROR   43              ;ERROR: ADDRESS COULD NOT BE MADE A HIT

020450  012737  033142  000114  T12H08: MOV     #UPERR,@#PVEC   ;RESTORE PARITY ERROR HANDLER



                                        ;;*************************************************************
                                        ;*TEST 25       TEST TAG ADDRESS BITS FOR HIGH HALF OF CACHE
                                        ;*
                                        ;*   THE TEST OF THE TAG BITS IS NOT COMPLETE UNTIL THE
                                        ;*TAG ADDRESS TEST FOR THE OTHER HALF OF CACHE AND THE
                                        ;*TEST OF THE MSB ADDRESS (A10) TO THE CACHE TAG FIELD
                                        ;*ARE RUN.  A WRITE/READ PROCEDURE IS DONE WHICH CHECKS
                                        ;*THE TAG FIELD BITS AND DUAL ADDRESSING ON THEM FOR HALF
                                        ;*OF CACHE.  MEMORY IS FIRST SIZED TO DETERMINE THE MAX-
                                        ;*IMUM TESTABLE ADDRESS.  THE TAG ADDRESS BITS OF THIS
                                        ;*ADDRESS ARE USED AS PATTERN A AND STORED IN KIPAR4.  A
                                        ;*PATTERN B IS NOW GENERATED WHICH HAS 'COMPLEMENT' TAG
                                        ;*BITS AND STORED IN KIPAR5.  ON THE FIRST PASS, PATTERN
                                        ;*A IS WRITTEN THROUGH HALF OF CACHE.  NEXT, STARTING AT
                                        ;*THE HIGH HALF CACHE ADDRESS, THE LOCATION IS READ,
                                        ;*CHECKED TO BE A HIT AND THEN WRITTEN WITH PATTERN B.
                                        ;*THIS IS SEQUENTIALLY REPEATED WITH DECREASING ADDRESSES
                                        ;*UNTIL THE LOW HALF CACHE ADDRESS IS REACHED.  AT THE
                                        ;*LOW ADDRESS, THE SECOND PATTERN IS READ, CHECKED TO BE A
                                        ;*HIT AND REWRITTEN WITH THE FIRST PATTERN.  THIS IS SE-
                                        ;*QUENTIALLY REPEATED WITH INCREASING ADDRESSES UNTIL THE
                                        ;*HIGH HALF CACHE ADDRESS IS REACHED.  A SECOND PASS IS
                                        ;*THEN MADE WITH THE PATTERNS REVERSED.
                                        ;*   ANY PARITY ERROR OR HIT ERROR IS REPORTED.
                                        ;*   DURING THE PASSES, R0, R1 CONTAIN ADDRESSES WHICH
                                        ;*REFERENCE KIPAR5,5.
                                        ;*   R3 INDICATES THE PASS NUMBER.
                                        ;*IF THE INHIBIT TESTS USING KT SWITCH (SW12) IS SET, THIS
                                        ;*TEST IS SKIPPED.

                                        ;;*************************************************************
020456  012737  000214  177746  TST25:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
020464  000004                          SCOPE
020466  012737  022000  001234          MOV     #TST26,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
020474  032777  010000  160437          BIT     #SW12,@SWR      ;INHIBIT TESTS USING KT?
020502  001402                          BEQ     3$              ;CONTINUE TEST IF NO
020504  000137  022000                  JMP     @#TST26         ;GO TO NEXT TEST
020510  052737  000200  036034  3$:     BIS     #200,@#$KT11    ;KT ON FOR $SIZE
020516  004737  035750                  JSR     PC,$SIZE        ;SIZE MEMORY
020522  012737  020744  000114          MOV     #T13H01,@#PVEC  ;SET UP PARITY ERROR HANDLER
020530  013737  036322  172350          MOV     @#$LSTBK,@#KIPAR4  ;SET UP PAR4 FOR ADDRESS PATTERN A

                                        ;CALC COMPLEMENT TAG PATTERN B

020536  013700  036322                  MOV     @#$LSTBK,R0     ;GET TEST PATTERN A AND
020542  005100                          COM     R0              ;CALC PATTERN B
020544  005001                          CLR     R1
```

```
020546  005201                  1$:     INC     R1
020550  006300                          ASL     R0
020552  100775                          BMI     1$
020554  006200                  2$:     ASR     R0
020556  077102                          SOB     R1,2$
020560  042700  000037                  BIC     #37,R0          ;ONLY COMPLEMENT TAG ADDR. BITS

020564  010037  172352                  MOV     R0,@#KIPAR5     ;SET UP PAR5 FOR ADDRESS PATTERN B

020570  012700  102000                  MOV     #102000,R0      ;INIT R0 TO ADDR PATTERN A
020574  012701  124000                  MOV     #124000,R1      ;INIT R1 TO ADDR PATTERN B
020600  005003                          CLR     R3              ;INIT FLAG FOR PASS 1
020602  005004                  T13H02: CLR     R4              ;INIT INDICATOR FOR ERROR LOOP 1
020604  012702  001000                  MOV     #1000,R2        ;INIT ADDR. COUNTER
020610  052737  000001  177572          BIS     #1,@#MMR0       ;TURN KT ON
020616  012737  000204  177746          MOV     #204,@#CCR      ;TURN HALF OF CACHE ON

020624  005720                  1$:     TST     (R0)+           ;WRITE PATTERN IN CACHE
020626  077202                          SOB     R2,1$           ;ALL DONE? BRANCH IF NO

020630  012702  001000                  MOV     #1000,R2        ;INIT. ADDR. COUNTER
020634  005740                  T13H03: TST     -(R0)           ;READ CACHE TAG BITS
020636  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT?
020644  001002                          BNE     2$              ;BRANCH  IF YES
020646  000137  021216                  JMP     T13H04          ;REPORT ERROR
020652  005741                  2$:     TST     -(R1)           ;WRITE NEW PATTERN IN TAG
020654  077211                          SOB     R2,T13H03                ;HALF ADDR. TESTED? BRANCH IF NO

020656  005204                          INC     R4              ;SET INDICATOR FOR ERROR LOOP 2
020660  012702  001000                  MOV     #1000,R2        ;INIT. ADDR. COUNTER
020664  005711                  T13H05: TST     (R1)            ;READ CACHE TAG BITS
020666  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT?
020674  001002                          BNE     3$              ;BRANCH IF YES
020676  000137  021264                  JMP     T13H06          ;REPORT ERROR
020702  005721                  3$:     TST     (R1)+           ;UPDATE FOR NEXT ADDRESS
020704  005720                          TST     (R0)+           ;WRITE NEW PATTERN IN TAG
020706  077212                          SOB     R2,T13H05

020710  005703                          TST     R3              ;SECOND PASS?
020712  001402                          BEQ     2$              ;CONTINUE TEST IF NO
020714  000137  021312                  JMP     T13H07          ;GO TO END OF TEST
020720  052703  000001          2$:     BIS     #1,R3           ;SET FLAG FOR SECOND PASS
020724  012737  020732  001110          MOV     #T13H15,@#$LPERR  ;INIT RETURN FOR ERROR LOOP IF ERROR OCCURS
020732  012700  122000          T13H15: MOV     #122000,R0      ;INIT. R0 TO ADDR. PATTERN B
020736  012701  104000                  MOV     #104000,R1      ;INIT. R1 TO ADDR. PATTERN A
020742  000717                          BR      T13H02          ;GO TEST SECOND PASS

020744  052737  000014  177746  T13H01: BIS     #14,@#CCR       ;CACHE OFF

020752  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
020754  076600                          MED                     ;GET CONTENTS OF LOG REG
020756  000022                          .WORD   RLOG
020760  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
020764  076600                          MED                     ;UNLOCK ERROR LOG
020766  000222                          .WORD   WLOG
020770  012600                          MOV     (SP)+,R0        ;RESTORE R0
```

```

020772  011637  001164                    MOV     (SP),$REG3        ;GET PC+2 OF TRAP
020776  162737  000002  001164            SUB     #2,$REG3          ;SAVE PC FOR MAIN PARITY ERROR
021004  022626                            CMP     (SP)+,(SP)+       ;RESTORE STACK
021006  010046                            MOV     R0,-(SP)          ;SAVE R0 ON STACK FOR MED INST.
021010  076600                            MED                       ;GET LOG INFOR FOR PHY. ADDR. A17,A16
021012  000101                            .WORD   RSER
021014  000300                            SWAB    R0                ;PUT PHY. ADDR A17, A16 IN LOW BYTE
021016  042700  177776                    BIC     #177776,R0        ;ONLY LOOK AT A17, A16
021022  010037  001160                    MOV     R0,$REG1          ;SAVE ADDRESS
021026  076600                            MED                       ;GET LOG INFORMATION
021030  000102                            .WORD   LOADD
021032  010037  001162                    MOV     R0,$REG2          ;SAVE INFORMATION
021036  076600                            MED                       ;GET LOG INFORMATION
021040  000100                            .WORD   RJAM
021042  012600                            MOV     (SP)+,R0          ;RESTORE R0
021044  032700  000400                    BIT     #400,R0           ;ERROR BACKING STORE?
021050  001402                            BEQ     T13H00            ;BRANCH IF NO
021052  104001                            ERROR   1                 ;ERROR: UNEXPECT. PARITY ERROR IN BACKING STORE
021054  000516                            BR      T13H07            ;GO TO END OF TEST

021056  011137  001166           T13H00:  MOV     (R1),$REG4        ;SAVE GOOD DATA
021062  005704                            TST     R4                ;ERROR IN LOOP 2?
021064  001002                            BNE     T13H09            ;BRANCH IF YES
021066  011037  001166                    MOV     (R0),$REG4        ;SAVE GOOD DATA

021072  032737  000040  177744   T13H09:  BIT     #40,@#EREG        ;TAG PARITY ERROR?
021100  001426                            BEQ     T13H10            ;BRANCH IF NO
021102  004737  033634                    JSR     PC,PAR            ;GET PAR USED
021106  000000                            .WORD   0                 ;INDICATOR FOR R0
021110  005704                            TST     R4                ;ERROR FROM LOOP 1?
021112  001403                            BEQ     T13H11            ;BRANCH IF YES
021114  004737  033634                    JSR     PC,PAR            ;GET PAR USED
021120  000001                            .WORD   1                 ;INDICATOR FOR R1
021122  004737  033606           T13H11:  JSR     PC,TAG            ;CALC TAG CONTENTS
021126  013737  001172  001166            MOV     $TMP0,$REG4       ;SAVE GOOD DATA
021134  076600                            MED                       ;GET TAG LOG INFO.
021136  000107                            .WORD   RTAG
021140  000300                            SWAB    R0                ;PUT TAG IN LOW BYTE
021142  042700  177400                    BIC     #177400,R0        ;LOOK AT TAG ONLY
021146  010037  001164                    MOV     R0,$REG3          ;SAVE BAD DATA
021152  104052                            ERROR   52                ;ERROR: TAG PARITY ERROR ON TEST OF TAG ADDRESS BITS
021154  000456                            BR      T13H07            ;GO TO END OF TEST

021156  032737  000100  177744   T13H10:  BIT     #100,@#EREG       ;LOW BYTE P.E.?
021164  001406                            BEQ     T13H12            ;BRANCH IF NO
021166  076600                            MED                       ;GET LOG INFORMATION
021170  000106                            .WORD   CDL
021172  010037  001164                    MOV     R0,$REG3          ;SAVE INFORMATION
021176  104053                            ERROR   53                ;ERROR: LOW BYTE PARITY ERROR ON TEST OF TAG ADDR. BITS
021200  000444                            BR      T13H07            ;GO TO END OF TEST

021202                           T13H12:
021202  076600                            MED                       ;GET LOG INFORMATION
021204  000106                            .WORD   CDH
021206  010037  001164                    MOV     R0,$REG3          ;SAVE INFORMATION
```

```
021212  104054                            ERROR   54                ;ERROR: HIGH BYTE PARITY ERROR ON TEST OF TAG ADDR. BITS
021214  000436                            BR      T13H07            ;GO TO END OF TEST

021216  052737  000014  177746   T13H04:  BIS     #14,@#CCR         ;CACHE OFF
021224  010037  001172                    MOV     R0,$TMP0          ;GET VIRTUAL ADDRESS TESTED
021230  004737  033434                    JSR     PC,VIP            ;SAVE ADDRESS TESTED
021234  062700  000002                    ADD     #2,R0             ;ADJUST ADDRESS WHEN LOOP
021240  004737  033634                    JSR     PC,PAR            ;GET PAR TESTED
021244  000000                            .WORD   0                 ;INDICATOR FOR R0
021246  004737  033606           T13H13:  JSR     PC,TAG            ;CALC TAG FROM PAR
021252  013737  001172  001164            MOV     $TMP0,$REG3       ;SAVE TAG
021260  104055                            ERROR   55                ;ERROR: TEST OF TAG ADDRESS BITS FAILED
                                                                    ;        ADDR. COULD NOT BE MADE A HIT
021262  000413                            BR      T13H07            ;GO TO NEXT TEST

021264  052737  000014  177746   T13H06:  BIS     #14,@#CCR         ;CACHE OFF

021272  010137  001172                    MOV     R1,$TMP0          ;GET VIRTUAL ADDRESS TESTED
021276  004737  033434                    JSR     PC,VIP            ;SAVE PHYSICAL ADDRESS TESTED
021302  004737  033634                    JSR     PC,PAR            ;GET PAR TESTED
021306  000001                            .WORD   1                 ;INDICATOR FOR R1
021310  000756                            BR      T13H13            ;REPORT ERROR

021312  005037  177572           T13H07:  CLR     @#MMR0            ;KT OFF
021316  012737  033142  000114            MOV     #UPERR,@#PVEC     ;RESTORE UNEXPECTED PARITY ERROR HANDLER
021324  052737  000014  177746            BIS     #14,@#CCR         ;CACHE OFF WHEN CROSS CACHE ADDR. BOUNDARY
021332  000137  022000                    JMP     @#TST26           ;GO TO NEXT TEST


        022000                   .=22000                            ;ADJUST ADDRESS SPACE FOR NEXT TEST



                                 ;;*********************************************************************
                                 ;*TEST 26        TEST DATA FIELD FOR LOW HALF OF CACHE
                                 ;*
                                 ;*   THE TEST OF THE DATA FIELD IS NOT COMPLETE UNTIL THE
                                 ;*TEST OF THE DATA FIELD FOR THE OTHER HALF OF CACHE AND
                                 ;*THE TEST OF THE MSB ADDRESS (A10) TO THE CACHE DATA
                                 ;*FIELD ARE RUN.  A WRITE/READ PROCEDURE IS DONE WHICH
                                 ;*CHECKS ALL THE DATA FIELD BITS AND DUAL ADDRESSING ON
                                 ;*THEM FOR HALF OF CACHE.  ON THE FIRST PASS ONE PATTERN
                                 ;*(CONTAINED IN R0) IS WRITTEN IN ALL THE DATA FIELDS,
                                 ;*FOR HALF OF CACHE.  NEXT, STARTING AT THE HIGH HALF
                                 ;*CACHE ADDRESS, THE LOCATION IS TESTED TO BE A HIT, ITS
                                 ;*DATA IS CHECKED AND THEN WRITTEN WITH A SECOND PATTERN
                                 ;*CONTAINED IN R1.  THIS IS SEQUENTIALLY REPEATED WITH
                                 ;*DECREASING ADDRESSES UNTIL THE LOW HALF CACHE ADDRESS IS
                                 ;*REACHED.  AT THE LOW ADDRESS, THE SECOND PATTERN IS READ,
                                 ;*TESTED TO BE A HIT AND REWRITTEN WITH THE FIRST PATTERN.
                                 ;*THIS IS SEQUENTIALLY REPEATED WITH INCREASING ADDRESSES
                                 ;*UNTIL THE HIGH HALF CACHE ADDRESS IS REACHED.  A SECOND
                                 ;*PASS IS THEN MADE WITH THE PATTERNS REVERSED.
                                 ;*   ANY PARITY RERR OR HIT ERROR IS REPORTED.
                                 ;*   R0, R1      CONTAIN THE TEST PATTERN
                                 ;*   R2          CONTAINS THE TEST ADDRESS
                                 ;*   R4          CONTAINS THE PASS NUMBER
```

```
                                         ;;*****************************************************************
ØØ22ØØØ  Ø12737  ØØØ214  177746  TST26:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
Ø22ØØ6   ØØØØØ4                          SCOPE
Ø22Ø1Ø   Ø12737  Ø24ØØØ  ØØ1234          MOV     #TST27,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
Ø22Ø16   Ø12737  Ø2221Ø  ØØØ114          MOV     #T15LØ1,@#PVEC  ;SET UP PARITY ERROR HANDLER
Ø22Ø24   Ø127ØØ  125252                  MOV     #125252,RØ      ;SET UP DATA PATTERN A FOR PASS 1
Ø22Ø3Ø   Ø127Ø1  Ø52525                  MOV     #52525,R1       ;SET UP DATA PATTERN B FOR PASS 1
Ø22Ø34   Ø12737  ØØØ21Ø  177746  T15LØ5: MOV     #21Ø,@#CCR      ;HALF CACHE ON
Ø22Ø42   ØØ5ØØ4                          CLR     R4              ;SET UP LOOP INDIC FOR ERROR LOOP 1
Ø22Ø44   Ø127Ø2  Ø6ØØØØ                  MOV     #BUFL,R2        ;INIT STARTING TEST ADDRESS
Ø22Ø5Ø   Ø127Ø3  ØØ1ØØØ                  MOV     #1ØØØ,R3        ;INIT ADDRESS COUNTER
Ø22Ø54   Ø1ØØ22                  1$:     MOV     RØ,(R2)+        ;WRITE CACHE WITH PATTERN
Ø22Ø56   Ø773Ø2                          SOB     R3,1$           ;LOOP TILL HALF CACHE WRITTEN

                                 ;NOW READ AND WRITE PATTERN, DECREASING ADDRESS

Ø22Ø6Ø   Ø127Ø3  ØØ1ØØØ                  MOV     #1ØØØ,R3        ;INIT ADDRESS COUNTER
Ø22Ø64   ØØ5742                  T15L21: TST     -(R2)           ;READ CACHE
Ø22Ø66   Ø33727  177752  ØØØØØ4          BIT     @#HMR,#HMR2     ;HIT?
Ø22Ø74   ØØ1ØØ2                          BNE     1$              ;BRANCH IF YES
Ø22Ø76   ØØØ137  Ø22466                  JMP     T15LØ2          ;REPORT ERROR
Ø221Ø2   Ø212ØØ                  1$:     CMP     (R2),RØ         ;IS DATA CORRECT?
Ø221Ø4   ØØ14Ø2                          BEQ     T15L17          ;BRANCH IF YES
Ø221Ø6   ØØØ137  Ø2251Ø                  JMP     T15LØ3          ;REPORT ERROR
Ø22112   Ø1Ø112                  T15L17: MOV     R1,(R2)         ;WRITE NEW PATTERN IN CACHE
Ø22114   Ø77315                          SOB     R3,T15L21       ;LOOP TILL HALF CACHE READ & WRITTEN

                                 ;NOW READ AND WRITE PATTERN, INCREASING ADDRESS

Ø22116   Ø527Ø4  ØØØØØ1                  BIS     #1,R4           ;SET FLAG FOR ERROR LOOP 2
Ø22122   Ø127Ø3  ØØ1ØØØ                  MOV     #1ØØØ,R3        ;INIT. ADDRESS COUNTER
Ø22126   ØØ5712                  T15L22: TST     (R2)            ;READ CACHE
Ø2213Ø   Ø33727  177752  ØØØØØ4          BIT     @#HMR,#HMR2     ;HIT?
Ø22136   ØØ1ØØ2                          BNE     1$              ;BRANCH IF YES
Ø2214Ø   ØØØ137  Ø22466                  JMP     T15LØ2          ;REPORT ERROR
Ø22144   Ø212Ø1                  1$:     CMP     (R2),R1         ;DATA OK?
Ø22146   ØØ14Ø2                          BEQ     T15L18          ;BRANCH IF YES
Ø2215Ø   ØØØ137  Ø22526                  JMP     T15L15          ;REPORT ERROR
Ø22154   Ø1ØØ22                  T15L18: MOV     RØ,(R2)+        ;WRITE NEW TEST PATTERN
Ø22156   Ø77315                          SOB     R3,T15L22       ;LOOP TILL HALF OF CACHE READ & WRITTEN

Ø2216Ø   ØØ57ØØ                          TST     RØ              ;DOES RØ HAVE DATA FOR FIRST PASS?
Ø22162   1ØØ4Ø2                          BMI     T15L12          ;BRANCH IF YES
Ø22164   ØØØ137  Ø2256Ø                  JMP     T15LØ4          ;GO TO END OF TEST
Ø2217Ø   Ø127ØØ  Ø52525          T15L12: MOV     #52525,RØ       ;SET UP DATA PATTERN B FOR PASS 2.
Ø22174   Ø127Ø1  125252                  MOV     #125252,R1      ;SET UP DATA PATTERN A FOR PASS 2
Ø222ØØ   Ø12737  Ø2217Ø  ØØ111Ø          MOV     #T15L12,@#$LPERR  ;INIT RETURN FOR ERROR LOOP IF ERROR
Ø222Ø6   ØØØ712                          BR      T15LØ5          ;GO TEST IT

Ø2221Ø   Ø52737  ØØØØ14  177746  T15LØ1: BIS     #14,@#CCR       ;CACHE OFF

Ø22216   Ø1ØØ46                          MOV     RØ,-(SP)        ;SAVE RØ FOR MED INST
Ø2222Ø   Ø766ØØ                          MED                     ;GET CONTENTS OF LOG REG
Ø22222   ØØØØ22                          .WORD   RLOG
Ø22224   Ø527ØØ  1ØØØØ1                  BIS     #1ØØØØ1,RØ      ;ENABLE ERROR LOG & LOG FIRST MODE
```

```
Ø2223Ø   Ø766ØØ                          MED                     ;UNLOCK ERROR LOG
Ø22232   ØØØ222                          .WORD   WLOG
Ø22234   Ø126ØØ                          MOV     (SP)+,RØ        ;RESTORE RØ

Ø22236   Ø11637  ØØ1164                  MOV     (SP),$REG3      ;GET PC+2 OF PARITY ERROR
Ø22242   162737  ØØØØØ2  ØØ1164          SUB     #2,$REG3        ;SAVE PC OF PARITY ERROR
Ø2225Ø   Ø22626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
Ø22252   Ø1ØØ46                          MOV     RØ,-(SP)        ;SAVE RØ FOR MED INST
Ø22254   Ø766ØØ                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
Ø22256   ØØØ1Ø1                          .WORD   RSER
Ø2226Ø   ØØØ3ØØ                          SWAB    RØ              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
Ø22262   Ø427ØØ  177776                  BIC     #177776,RØ      ;ONLY LOOK AT A17, A16
Ø22266   Ø1ØØ37  ØØ116Ø                  MOV     RØ,$REG1        ;SAVE ADDRESS
Ø22272   Ø766ØØ                          MED                     ;GET LOG INFORMATION
Ø22274   ØØØ1Ø2                          .WORD   LOADD
Ø22276   Ø1ØØ37  ØØ1162                  MOV     RØ,$REG2        ;SAVE INFORMATION
Ø223Ø2   Ø766ØØ                          MED                     ;GET LOG INFORMATION
Ø223Ø4   ØØØ1ØØ                          .WORD   RJAM
Ø223Ø6   Ø1ØØØ5                          MOV     RØ,R5   ;SAVE INFORMATION
Ø2231Ø   Ø126ØØ                          MOV     (SP)+,RØ        ;RESTORE RØ
Ø22312   Ø327Ø5  ØØØ4ØØ                  BIT     #4ØØ,R5         ;ERROR IN BACKING STORE?
Ø22316   ØØ14Ø6                          BEQ     T15LØ6          ;BRANCH IF NO
Ø2232Ø   Ø766ØØ                          MED                     ;GET LOG INFORMATION
Ø22322   Ø55Ø16                          .WORD   BSD
Ø22324   Ø1ØØ37  ØØ1164                  MOV     RØ,$REG3        ;SAVE INFORMATION
Ø2233Ø   1Ø4ØØ1                          ERROR   1               ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE
Ø22332   ØØØ512                          BR      T15LØ4          ;GO TO END OF TEST

Ø22334   Ø1Ø137  ØØ1166          T15LØ6: MOV     R1,$REG4        ;SAVE GOOD DATA
Ø2234Ø   ØØ57Ø4                          TST     R4              ;ERROR LOOP 1?
Ø22342   ØØ1ØØ2                          BNE     T15LØ8          ;BRANCH IF NO
Ø22344   Ø1ØØ37  ØØ1166                  MOV     RØ,$REG4        ;SAVE GOOD DATA

Ø2235Ø   Ø32737  ØØØ1ØØ  177744  T15LØ8: BIT     #1ØØ,@#EREG     ;LOW BYTE PARITY ERROR?
Ø22356   ØØ14Ø6                          BEQ     T15L13          ;BRANCH IF NO
Ø2236Ø   Ø766ØØ                          MED                     ;GET LOG INFORMATION
Ø22362   ØØØ1Ø6                          .WORD   CDL
Ø22364   Ø1ØØ37  ØØ1164                  MOV     RØ,$REG3        ;SAVE INFORMATION
Ø2237Ø   1Ø4Ø56                          ERROR   56              ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA FIELD
Ø22372   ØØØ472                          BR      T15LØ4          ;GO TO END OF TEST

Ø22374   Ø32737  ØØØ2ØØ  177744  T15L13: BIT     #2ØØ,@#EREG     ;PARITY ERROR IN HIGH BYTE?
Ø224Ø2   ØØ14Ø6                          BEQ     T15L14          ;BRANCH IF NO
Ø224Ø4   Ø766ØØ                          MED                     ;GET LOG INFORMATION
Ø224Ø6   ØØØ1Ø6                          .WORD   CDH
Ø2241Ø   Ø1ØØ37  ØØ1164                  MOV     RØ,$REG3        ;SAVE INFORMATION
Ø22414   1Ø4Ø57                          ERROR   57              ;ERROR: HIGH BYTE PARITY ERROR WHEN TEST DATA FIELD
Ø22416   ØØØ46Ø                          BR      T15LØ4          ;GO TO END OF TEST

Ø2242Ø   Ø1Ø237  ØØ1166          T15L14: MOV     R2,$REG4        ;GET FAILING ADDRESS
Ø22424   Ø127Ø5  ØØØØ13                  MOV     #13,R5          ;SET UP COUNTER
Ø2243Ø   ØØ6237  ØØ1166          2$:     ASR     $REG4           ;PUT TAG ADDRESS BITS IN LSB 6-Ø
Ø22434   Ø775Ø3                          SOB     R5,2$           ;LOOP TILL DONE
Ø22436   Ø52737  ØØØ2ØØ  ØØ1166          BIS     #2ØØ,$REG4      ;SET VALID BIT
Ø22444   Ø766ØØ                          MED                     ;GET TAG LOG INFO.
Ø22446   ØØØ1Ø7                          .WORD   RTAG
```

```
022450  000300                           SWAB    R0              ;PUT TAG IN LOW BYTE
022452  042700  177400                   BIC     #177400,R0      ;LOOK AT TAG ONLY
022456  010037  001164                   MOV     R0,$REG3        ;SAVE BAD DATA
022462  104060                           ERROR   60              ;ERROR: TAG PARITY ERROR WHEN TESTING DATA FIELD
022464  000435                           BR      T15L04          ;GO TO END OF TEST

022466  052737  000014  177746  T15L02:  BIS     #14,@#CCR       ;CACHE OFF
022474  005037  001160                   CLR     $REG1           ;SAVE ADDRESS
022500  010237  001162                   MOV     R2,$REG2        ;SAVE ADDRESS
022504  104043                           ERROR   43              ;ERROR: ADDRESS COULD NOT BE MADE A HIT
022506  000424                           BR      T15L04          ;GO TO END OF TEST

022510  011205                  T15L03:  MOV     (R2),R5         ;GET BAD DATA
022512  052737  000014  177746           BIS     #14,@#CCR       ;CACHE OFF
022520  010037  001166                   MOV     R0,$REG4        ;SAVE GOOD DATA
022524  000406                           BR      T15L16          ;REPORT ERROR

022526  011205                  T15L15:  MOV     (R2),R5         ;GET BAD DATA
022530  052737  000014  177746           BIS     #14,@#CCR       ;CACHE OFF
022536  010137  001166                   MOV     R1,$REG4        ;SAVE GOOD DATA
022542  005037  001160          T15L16:  CLR     $REG1           ;SAVE ADDRESS
022546  010237  001162                   MOV     R2,$REG2        ;SAVE ADDRESS
022552  010537  001164                   MOV     R5,$REG3        ;SAVE BAD DATA
022556  104061                           ERROR   61              ;ERROR: CACHE DATA LOC HELD WRONG DATA

022560  012737  033142  000114  T15L04:  MOV     #UPERR,@#PVEC   ;RESTORE UNEXPECT. P.E. HANDLER
022566  052737  000014  177746           BIS     #14,@#CCR       ;CACHE OFF WHEN CROSS CACHE ADDR. BOUNDARY
022574  000137  024000                   JMP     @#TST27         ;GO TO NEXT TEST


        024000                           .=24000                 ;ADJUST ADDRESS SPACE FOR NEXT TEST




                                ;;*************************************************************
                                ;*TEST 27        TEST DATA FIELD FOR HIGH HALF OF CACHE
                                ;*
                                ;*   THE TEST OF THE DATA FIELD IS NOT COMPLETE UNTIL THE
                                ;*TEST OF THE DATA FIELD FOR THE OTHER HALF OF CACHE AND
                                ;*THE TEST OF THE MSB ADDRESS (A10) TO THE CACHE DATA
                                ;*FIELD ARE RUN.  A WRITE/READ PROCEDURE IS DONE WHICH
                                ;*CHECKS ALL THE DATA FIELD BITS AND DUAL ADDRESSING ON
                                ;*THEM FOR HALF OF CACHE.  ON THE FIRST PASS ONE PATTERN
                                ;*(CONTAINED IN R0) IS WRITTEN IN ALL THE DATA FIELDS.
                                ;*FOR HALF OF CACHE.  NEXT, STARTING AT THE HIGH HALF
                                ;*CACHE ADDRESS, THE LOCATION IS TESTED TO BE A HIT, ITS
                                ;*DATA IS CHECKED AND THEN WRITTEN WITH A SECOND PATTERN
                                ;*CONTAINED IN R1.  THIS IS SEQUENTIALLY REPEATED WITH
                                ;*DECREASING ADDRESSES UNTIL THE LOW HALF CACHE ADDRESS IS
                                ;*REACHED.  AT THE LOW ADDRESS, THE SECOND PATTERN IS READ,
                                ;*TESTED TO BE A HIT AND REWRITTEN WITH THE FIRST PATTERN.
                                ;*THIS IS SEQUENTIALLY REPEATED WITH INCREASING ADDRESSES
                                ;*UNTIL THE HIGH HALF CACHE ADDRESS IS REACHED.  A SECOND
                                ;*PASS IS THEN MADE WITH THE PATTERNS REVERSED.
                                ;*   ANY PARITY REEOR OR HIT ERROR IS REPORTED.
                                ;*   R0, R1      CONTAIN THE TEST PATTERN
```

```
                                ;*   R2          CONTAINS THE TEST ADDRESS
                                ;*   R4          CONTAINS THE PASS NUMBER

                                ;;*************************************************************
024000  012737  000214  177746  TST27:   MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
024006  000004                           SCOPE
024010  012737  024566  001234           MOV     #TST30,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
024016  012737  024210  000114           MOV     #T15H01,@#PVEC  ;SET UP PARITY ERROR HANDLER
024024  012700  125252                   MOV     #125252,R0      ;SET UP DATA PATTERN A FOR PASS 1
024030  012701  052525                   MOV     #52525,R1       ;SET UP DATA PATTERN B FOR PASS 1
024034  012737  000204  177746  T15H05:  MOV     #204,@#CCR      ;HALF CACHE ON
024042  005004                           CLR     R4              ;SET UP LOOP INDIC FOR ERROR LOOP 1
024044  012702  062000                   MOV     #BUFH,R2        ;INIT STARTING TEST ADDRESS
024050  012703  001000                   MOV     #1000,R3        ;INIT ADDRESS COUNTER
024054  010022                  1$:      MOV     R0,(R2)+        ;WRITE CACHE WITH PATTERN
024056  077302                           SOB     R3,1$           ;LOOP TILL HALF CACHE WRITTEN

                                ;NOW READ AND WRITE PATTERN, DECREASING ADDRESS

024060  012703  001000                   MOV     #1000,R3        ;INIT ADDRESS COUNTER
024064  005742                  T15H21:  TST     -(R2)           ;READ CACHE
024066  033727  177752  000004           BIT     @#HMR,#HMR2     ;HIT?
024074  001002                           BNE     1$              ;BRANCH IF YES
024076  000137  024466                   JMP     T15H02          ;REPORT ERROR
024102  021200                  1$:      CMP     (R2),R0         ;IS DATA CORRECT?
024104  001402                           BEQ     T15H17          ;BRANCH IF YES
024106  000137  024510                   JMP     T15H03          ;REPORT ERROR
024112  010112                  T15H17:  MOV     R1,(R2)         ;WRITE NEW PATTERN IN CACHE
024114  077315                           SOB     R3,T15H21       ;LOOP TILL HALF CACHE READ & WRITTEN

                                ;NOW READ AND WRITE PATTERN, INCREASING ADDRESS

024116  052704  000001                   BIS     #1,R4           ;SET FLAG FOR ERROR LOOP 2
024122  012703  001000                   MOV     #1000,R3        ;INIT. ADDRESS COUNTER
024126  005712                  T15H22:  TST     (R2)            ;READ CACHE
024130  033727  177752  000004           BIT     @#HMR,#HMR2     ;HIT?
024136  001002                           BNE     1$              ;BRANCH IF YES
024140  000137  024466                   JMP     T15H02          ;REPORT ERROR
024144  021201                  1$:      CMP     (R2),R1         ;DATA OK?
024146  001402                           BEQ     T15H18          ;BRANCH IF YES
024150  000137  024526                   JMP     T15H15          ;REPORT ERROR
024154  010022                  T15H18:  MOV     R0,(R2)+        ;WRITE NEW TEST PATTERN
024156  077315                           SOB     R3,T15H22       ;LOOP TILL HALF OF CACHE READ & WRITTEN

024160  005700                           TST     R0              ;DOES R0 HAVE DATA FOR FIRST PASS?
024162  100402                           BMI     T15H12          ;BRANCH IF YES
024164  000137  024560                   JMP     T15H04          ;GO TO END OF TEST
024170  012700  052525          T15H12:  MOV     #52525,R0       ;SET UP DATA PATTERN B FOR PASS 2.
024174  012701  125252                   MOV     #125252,R1      ;SET UP DATA PATTERN A FOR PASS 2
024200  012737  024170  001110           MOV     #T15H12,@#$LPERR  ;INIT RETURN FOR ERROR LOOP IF ERROR
024206  000712                           BR      T15H05          ;GO TEST IT

024210  052737  000014  177746  T15H01:  BIS     #14,@#CCR       ;CACHE OFF

024216  010046                           MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
024220  076600                           MED                     ;GET CONTENTS OF LOG REG
```

```
024222  000022                          .WORD   RLOG
024224  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
024230  076600                          MED                     ;UNLOCK ERROR LOG
024232  000222                          .WORD   WLOG
024234  012600                          MOV     (SP)+,R0        ;RESTORE R0

024236  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 OF PARITY ERROR
024242  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF PARITY ERROR
024250  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
024252  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
024254  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR, A17,A16
024256  000101                          .WORD   RSER
024260  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
024262  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
024266  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
024272  076600                          MED                     ;GET LOG INFORMATION
024274  000102                          .WORD   LOADD
024276  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
024302  076600                          MED                     ;GET LOG INFORMATION
024304  000100                          .WORD   RJAM
024306  010005                          MOV     R0,R5   ;SAVE INFORMATION
024310  012600                          MOV     (SP)+,R0        ;RESTORE R0
024312  032705  000400                  BIT     #400,R5         ;ERROR IN BACKING STORE?
024316  001406                          BEQ     T15H06          ;BRANCH IF NO
024320  076600                          MED                     ;GET LOG INFORMATION
024322  055P16                          .WORD   BSD
024324  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
024330  104001                          ERROR   1               ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE
024332  000512                          BR      T15H04          ;GO TO END OF TEST

024334  010137  001166          T15H06: MOV     R1,$REG4        ;SAVE GOOD DATA
024340  005704                          TST     R4              ;ERROR LOOP 1?
024342  001002                          BNE     T15H08          ;BRANCH IF NO
024344  010037  001166                  MOV     R0,$REG4        ;SAVE GOOD DATA

024350  032737  000100  177744  T15H08: BIT     #100,@#EREG     ;LOW BYTE PARITY ERROR?
024356  001406                          BEQ     T15H13          ;BRANCH IF NO
024360  076600                          MED                     ;GET LOG INFORMATION
024362  000106                          .WORD   CDL
024364  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
024370  104056                          ERROR   56              ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA FIELD
024372  000472                          BR      T15H04          ;GO TO END OF TEST

024374  032737  000200  177744  T15H13: BIT     #200,@#EREG     ;PARITY ERROR IN HIGH BYTE?
024402  001406                          BEQ     T15H14          ;BRANCH IF NO
024404  076600                          MED                     ;GET LOG INFORMATION
024406  000106                          .WORD   CDH
024410  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
024414  104057                          ERROR   57              ;ERROR: HIGH BYTE PARITY ERROR WHEN TEST DATA FIELD
024416  000460                          BR      T15H04          ;GO TO END OF TEST

024420  010237  001166          T15H14: MOV     R2,$REG4        ;GET FAILING ADDRESS
024424  012705  000013                  MOV     #13,R5          ;SET UP COUNTER
024430  006237  001166          20:     ASR     $REG4           ;PUT TAG ADDRESS BITS IN LSB 6-0
024434  077503                          SOB     R5,20           ;LOOP TILL DONE
024436  052737  000200  001166          BIS     #200,$REG4      ;SET VALID BIT
```

```
024444  076600                          MED                     ;GET TAG LOG INFO.
024446  000107                          .WORD   RTAG
024450  000300                          SWAB    R0              ;PUT TAG IN LOW BYTE
024452  042700  177400                  BIC     #177400,R0      ;LOOK AT TAG ONLY
024456  010037  001164                  MOV     R0,$REG3        ;SAVE BAD DATA
024462  104060                          ERROR   60              ;ERROR: TAG PARITY ERROR WHEN TESTING DATA FIELD
024464  000435                          BR      T15H04          ;GO TO END OF TEST

024466  052737  000014  177746  T15H02: BIS     #14,@#CCR       ;CACHE OFF
024474  005037  001160                  CLR     $REG1           ;SAVE ADDRESS
024500  010237  001162                  MOV     R2,$REG2        ;SAVE ADDRESS
024504  104043                          ERROR   43              ;ERROR: ADDRESS COULD NOT BE MADE A HIT
024506  000424                          BR      T15H04          ;GO TO END OF TEST

024510  011205                  T15H03: MOV     (R2),R5         ;GET BAD DATA
024512  052737  000014  177746          BIS     #14,@#CCR       ;CACHE OFF
024520  010037  001166                  MOV     R0,$REG4        ;SAVE GOOD DATA
024524  000406                          BR      T15H16          ;REPORT ERROR

024526  011205                  T15H15: MOV     (R2),R5         ;GET BAD DATA
024530  052737  000014  177746          BIS     #14,@#CCR       ;CACHE OFF
024536  010137  001166                  MOV     R1,$REG4        ;SAVE GOOD DATA
024542  005037  001160          T15H16: CLR     $REG1           ;SAVE ADDRESS
024546  010237  001162                  MOV     R2,$REG2        ;SAVE ADDRESS
024552  010537  001164                  MOV     R5,$REG3        ;SAVE BAD DATA
024556  104061                          ERROR   61              ;ERROR: CACHE DATA LOC HELD WRONG DATA

024560  012737  033142  000114  T15H04: MOV     #UPERR,@#PVEC   ;RESTORE UNEXPEC. PARITY ERROR HANDLER

                                ;;*****************************************************************
                                ;*TEST 30        TEST OF MSB ADDRESS (A10) TO VALID BIT
                                ;*
                                ;*      THIS IS THE FIRST TEST WHERE ALL OF CACHE IS TURNED
                                ;*ON.  THE TEST CHECKS FOR DUAL ADDRESSING ON THE VALID BIT FOR
                                ;*THE MSB PHYSICAL ADDRESS (A10) TO CACHE.  INITIALLY TEST ADDRESSES
                                ;*ARE CHOSEN WHICH HAVE THE CACHE ADDRESS BITS A1-A9 THE SAME
                                ;*AND A10 COMPLEMENTS.  THE ADDRESSES ARE ALSO CHOSEN TO NOT OVERLAP
                                ;*THE TEST INSTRUCTION SPACE.  THE FIRST ADDRESS IS AT THE END OF THIS
                                ;*TEST INSTRUCTION SPACE (TAD2) AND THE SECOND IS CHOSEN BY THE
                                ;*SUBROUTINE HAD TO LIE IN A 1 K BUFFER AT THE END OF THE PROGRAM.
                                ;*      THE FIRST ADDRESS IS INVALIDATED VIA WWP AND FORCING A PARITY
                                ;*TRAP.  THE SECOND IS THEN MADE VALID AND CHECKED TO BE A HIT.  THE FIRST IS
                                ;*THEN EXAMINED TO STILL BE INVALID (NOT A HIT).  ANY PARITY OR HIT
                                ;*ERROR IS REPORTED.

                                ;;*****************************************************************
024566  012737  000214  177746  TST30:  MOV #214,@#CCR  ;CACHE OFF FOR SCOPE
024574  000004                          SCOPE
024576  012737  025244  001234          MOV     #TST31,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
024604  012737  024646  000114          MOV     #T30L01,@#PVEC  ;SET UP FOR PARITY TRAP
024612  004737  033714                  JSR     PC,HAD          ;CALC CONGRUENT ADDR. IN TEST BUFFER
024616  025242                          .WORD   TAD2
024620  013700  001172                  MOV     $TMP0,R0        ;SAVE ADDR.
024624  012737  000300  177746          MOV     #300,@#CCR      ;CACHE ON & WWP
024632  005010                          CLR     (R0)            ;WWP IN TEST ADDR.
024634  012737  000200  177746          MOV     #200,@#CCR      ;WWP OFF
```

```
024642  005710                          TST     (R0)            ;FORCE PARITY TRAP
024644  000465                          BR      T30L02          ;REPORT FAILURE TO TRAP

024646                          T30L01:

024646  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
024650  076600                          MED                     ;GET CONTENTS OF LOG REG
024652  000022                          .WORD   RLOG
024654  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
024660  076600                          MED                     ;UNLOCK ERROR LOG
024662  000222                          .WORD   WLOG
024664  012600                          MOV     (SP)+,R0        ;RESTORE R0

024666  062706  000004                  ADD     #4,SP           ;RESTORE STACK
024672  012737  025042  000114          MOV     #T30L06,@#PVEC  ;SET UP PARITY ERROR HANDLER
024700  023737  025242  025242          CMP     TAD2,TAD2       ;MAKE TEST ADDR A HIT
024706  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT?
024714  001427                          BEQ     T30L03          ;REPORT ERROR IF NO
024716  005710                          TST     (R0)            ;CHECK OTHER LOC. STILL INVALIDATED
024720  033727  177752  000004          BIT     @#HMR,#HMR2     ;MISS?
024726  001011                          BNE     T30L04          ;REPORT ERROR IF NO
024730                          T30L05:

                                        ;RID CACHE OF BAD PARITY
024730  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
024736  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE


024742  012737  033142  000114          MOV     #UPERR,@#PVEC   ;RESTORE UNEXP. PARITY ERROR HANDLER
024750  000535                          BR      TST31           ;;GO TO NEXT TEST

024752  012737  000214  177746  T30L04: MOV     #214,@#CCR      ;CACHE OFF
024760  005037  001160                  CLR     $REG1           ;SAVE BAD ADDRESS
024764  010037  001162                  MOV     R0,$REG2        ;SAVE BAD ADDRESS
024770  104121                          ERROR   121             ;ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED
                                                                ;       LOC. NOT INVALIDATED
024772  000756                          BR      T30L05          ;GO TO END OF TEST

024774  012737  000214  177746  T30L03: MOV     #214,@#CCR      ;CACHE OFF
025002  005037  001160                  CLR     $REG1           ;SAVE BAD ADDRESS
025006  012737  025242  001162          MOV     #TAD2,$REG2     ;SAVE BAD ADDRESS
025014  104043                          ERROR   43              ;ERROR:ADDRESS COULD NOT BE MADE A HIT
025016  000744                          BR      T30L05          ;GO TO END OF TEST

025020  012737  000214  177746  T30L02: MOV     #214,@#CCR      ;CACHE OFF
025026  005037  001160                  CLR     $REG1           ;SAVE BAD ADDRESS
025032  010037  001162                  MOV     R0,$REG2        ;SAVE BAD ADDRESS
025036  104042                          ERROR   42              ;ERROR:NO PARITY TRAP FROM LOC WRITTEN WITH WRONG PAR.
025040  000733                          BR      T30L05          ;GO TO END OF TEST

025042  012737  000214  177746  T30L06: MOV     #214,@#CCR      ;CACHE OFF

025050  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
025052  076600                          MED                     ;GET CONTENTS OF LOG REG
025054  000022                          .WORD   RLOG
025056  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
```

```
025062  076600                          MED                     ;UNLOCK ERROR LOG
025064  000222                          .WORD   WLOG
025066  012600                          MOV     (SP)+,R0        ;RESTORE R0

025070  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 OF ERROR
025074  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF ERROR
025102  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
025104  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
025106  000101                          .WORD   RSER
025110  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
025112  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
025116  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
025122  076600                          MED                     ;GET LOG INFORMATION
025124  000102                          .WORD   LOADD
025126  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
025132  076600                          MED                     ;GET LOG INFORMATION
025134  000100                          .WORD   RJAM
025136  032700  000400                  BIT     #400,R0         ;ERROR IN BACKING STORE?
025142  001402                          BEQ     1$              ;BRANCH IF NO
025144  104001                          ERROR   1               ;ERROR:UNEXP. PARITY ERROR IN BACKING STORE
025146  000670                          BR      T30L05          ;GO TO END OF TEST

025150  032737  000040  177744  1$:     BIT     #40,@#EREG      ;PARITY ERROR TAG?
025156  001411                          BEQ     2$              ;BRANCH IF NO
025160  076600                          MED                     ;GET TAG LOG INFO.
025162  000107                          .WORD   RTAG
025164  000300                          SWAB    R0              ;PUT TAG IN LOW BYTE
025166  042700  177400                  BIC     #177400,R0      ;LOOK AT TAG ONLY
025172  010037  001164                  MOV     R0,$REG3        ;SAVE BAD DATA
025176  104122                          ERROR   122             ;ERROR:TEST OF MSB ADDR. (A10) TO VALID BIT FAILED
                                                                ;       PARITY ERROR TAG
025200  000653                          BR      T30L05          ;GO TO END OF TEST

025202  032737  000100  177744  2$:     BIT     #100,@#EREG     ;PARITY ERROR LOW BYTE?
025210  001406                          BEQ     3$              ;BRANCH IF NO
025212  076600                          MED                     ;GET LOG INFORMATION
025214  000106                          .WORD   CDL
025216  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
025222  104123                          ERROR   123             ;ERROR:TEST OF MSB ADDR. (A10) TO VALID BIT FAILED
                                                                ;       PARITY ERROR LOW BYTE
025224  000641                          BR      T30L05          ;GO TO END OF TEST

025226                          3$:
025226  076600                          MED                     ;GET LOG INFORMATION
025230  000106                          .WORD   CDH
025232  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
025236  104124                          ERROR   124             ;ERROR:TEST OF MSB ADDR. (A10) TO VALID BIT FAILED
                                                                ;       PARITY ERROR HIGH BYTE
025240  000633                          BR      T30L05          ;GO TO END OF TEST

025242  000000                  TAD2:   .WORD   0               ;TEST ADDRESS


                                ;;***************************************************************
                                ;*TEST 31       TEST OF MSB ADDRESS (A10) TO CACHE TAG FIELD
                                ;*
                                ;*      THIS TEST CHECKS FOR DUAL ADDRESSING ON THE TAG
```

```
                                        ;*FIELD FOR THE MSB ADDRESS (A10) TO CACHE.  THERE ARE TWO
                                        ;*PASSES.  THE FIRST EXERCISES THE ADDRESS BITS IN THE TAG
                                        ;*FIELD AND THE SECOND EXERCISES THE TAG P BIT.  INITIALLY
                                        ;*THE MEMORY IS SIZED TO DETERMINE THE MAXIMUM TESTABLE
                                        ;*ADDRESS.  THE TAG FIELD OF THE MAX ADDR. IS USED AS THE
                                        ;*FIRST TEST VALUE AND ITS COMPLEMENT AS THE SECOND. THESE
                                        ;*TAG VALUES ARE THEN PUT INTO CACHE LOCATIONS WITH THE
                                        ;*SAME CACHE ADDRESS (A1-A9) EXCEPT FOR THEIR ADDRESS BIT
                                        ;*A10 COMPLEMENTS.  THE LOCS IN CACHE ARE CHOSEN SO THAT
                                        ;*THEY DON'T OVERLAP THE TEST INSTRUCTION ADDRESS SPACE.
                                        ;*THIS IS TO PREVENT THEIR BEING SWAPT OUT WHEN THE INSTRUC-
                                        ;*TIONS ARE BEING EXECUTED.  AFTER THE LOCATIONS ARE
                                        ;*WRITTEN THEY ARE EXAMINED AND CHECKED TO BE HITS.
                                        ;*FOLLOWING THIS THE SECOND PASS IS DONE FOR THE TAG P BIT.
                                        ;*TWO NEW TAG VALUES ARE CHOSEN WITH OPPOSITE P BITS.  THEY ARE
                                        ;*THEN WRITTEN, READ AND TESTED FOR HITS.  ANY PARITY ERRORS
                                        ;*OR HIT ERRORS ARE REPORTED.
                                        ;*   KIPAR4, 5 CONTAIN THE TAG VALUES WHICH ARE STORED IN
                                        ;*CACHE.
                                        ;*   R0, R1 CONTAIN THE CACHE TEST LOC THAT DON'T OVERLAP
                                        ;*THE INSTRUCTION ADDRESS SPACE
                                        ;*    R5 CONTAINS THE PASS #.
                                        ;*    IF THE INHIBIT TEST USING KT SWITCH (SW12) IS SET,
                                        ;*THIS TEST IS SKIPPED.

                                        ;;*************************************************************
 025244  012737  000214  177746  TST31:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
 025252  000004                          SCOPE
 025254  012737  025750  001234          MOV     #TST32,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 025262  032777  010000  153644          BIT     #SW12,@SWR      ;INHIBIT TESTS USING KT?
 025270  001402                          BEQ     3$              ;CONTINUE TEST IF NO
 025272  000137  025750                  JMP     @#TST32         ;GO TO NEXT TEST
 025276  012737  025534  000114  3$:     MOV     #T16L01,@#PVEC  ;SET UP PARITY ERROR HANDLER
 025304  052737  000200  036034          BIS     #200,@#$KT11    ;USE KT FOR $SIZE
 025312  004737  035750                  JSR     PC,$SIZE        ;SIZE MEMORY
 025316  013700  036322                  MOV     @#$LSTBK,R0     ;GET LAST ADDRESS AND
 025322  005100                          COM     R0              ;CALC. ITS COMPLEMENT
 025324  005001                          CLR     R1              ;KEEPING THE MSB THAT ARE 0
 025326  005201                  1$:     INC     R1              ;A 0
 025330  006300                          ASL     R0
 025332  100775                          BMI     1$
 025334  006200                  2$:     ASR     R0
 025336  077102                          SOB     R1,2$
 025340  042700  000037                  BIC     #37,R0          ;ONLY COMPLEMENT TAG ADDRESS BITS
 025344  010037  172352                  MOV     R0,@#KIPAR5     ;SET UP PAR5 WITH COMPLEMENT ADDRESS BITS
 025350  013737  036322  172350          MOV     @#$LSTBK,@#KIPAR4 ;SET UP PAR4 WITH COMPLEMENT ADDRESS BITS

                                         ;SET UP R0,R1 TO ADDR. LOCS WHICH DON'T OVERLAP THIS TEST'S INSTRUCTION SPACE

 025356  012700  025746                  MOV     #LAST1,R0       ;GET ADDR. OF LAST IN THIS TEST
 025362  042700  174000                  BIC     #174000,R0      ;SAVE LOWER ADDR BITS A10-A0
 025366  010001                          MOV     R0,R1           ;COPY ADDRESS
 025370  062700  100000                  ADD     #100000,R0      ;HAVE R0 ADDR PAR4
 025374  062701  122000                  ADD     #122000,R1      ;HAVE R1 ADDR PAR5 & HAVE A10 COMP OF R0
 025400  005005                          CLR     R5              ;INDICATE PASS 1
 025402  052737  000001  177572          BIS     #1,@#MMR0       ;KT ON
```

```
 025410  012737  000200  177746  T16L05: MOV     #200,@#CCR      ;CACHE ON
 025416  021011                          CMP     (R0),(R1)       ;GET LOC IN CACHE VIA DATI
 025420  005710                          TST     (R0)            ;READ CACHE
 025422  033727  177752  000004          BIT     @#HMR,#HMR2     ;SEE IF HIT
 025430  001425                          BEQ     T16L02          ;BRANCH IF NO TO ERROR
 025432  005711                          TST     (R1)            ;READ CACHE
 025434  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT?
 025442  001412                          BEQ     T16L03          ;BRANCH IF NO
 025444  005705                          TST     R5              ;FIRST PASS?
 025446  001131                          BNE     T16L04          ;BRANCH IF NO TO END OF TEST
 025450  052705  000001                  BIS     #1,R5           ;SET FLAG FOR SECOND PASS
 025454  005037  172350                  CLR     @#KIPAR4        ;SET UP PAR4 TO TEST P BIT
 025460  012737  000040  172352          MOV     #40,@#KIPAR5    ;SET UP PAR5 TO TEST P BIT
 025466  000750                          BR      T16L05          ;TEST IT

 025470  052737  000014  177746  T16L03: BIS     #14,@#CCR       ;CACHE OFF

 025476  010137  001172                  MOV     R1,$TMP0        ;GET VIRTUAL ADDRESS
 025502  000405                          BR      T16L06          ;CONVERT VIRTUAL INTO PHYSICAL ADDR

 025504  052737  000014  177746  T16L02: BIS     #14,@#CCR       ;CACHE OFF
 025512  010037  001172                  MOV     R0,$TMP0        ;GET VIRTUAL ADDR.
 025516  004737  033434          T16L06: JSR     PC,VIP          ;CHANGE VIRTUAL ADDRESS INTO PHYSICAL
 025522  012737  025410  001110          MOV     #T16L05,@#$LPERR ;SETUP RETURN FOR ERROR LOOP
 025530  104067                          ERROR   67              ;ERROR: TEST OF MSB ADDRESS (A10) TO TAG FIELD FAILEC
                                                                 ;        ADDRESS COULD NOT BE MADE A HIT
 025532  000477                          BR      T16L04          ;GO TO END OF TEST

 025534  052737  000014  177746  T16L01: BIS     #14,@#CCR       ;CACHE OFF

 025542  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
 025544  076600                          MED                     ;GET CONTENTS OF LOG REG
 025546  000022                          .WORD   RLOG
 025550  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
 025554  076600                          MED                     ;UNLOCK ERROR LOG
 025556  000222                          .WORD   WLOG
 025560  012600                          MOV     (SP)+,R0        ;RESTORE R0

 025562  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 OF ERROR
 025566  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF ERROR
 025574  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
 025576  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
 025600  000101                          .WORD   RSER
 025602  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
 025604  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
 025610  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
 025614  076600                          MED                     ;GET LOG INFORMATION
 025616  000102                          .WORD   LOADD
 025620  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
 025624  076600                          MED                     ;GET LOG INFORMATION
 025626  000100                          .WORD   RJAM
 025630  032700  000400                  BIT     #400,R0         ;ERROR IN BACKING STORE
 025634  001402                          BEQ     T16L07          ;BRANCH IF NO
 025636  104001                          ERROR   1               ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE
 025640  000434                          BR      T16L04          ;GO TO END OF TEST

 025642  032737  000040  177744  T16L07: BIT     #40,@#EREG      ;ERROR IN TAG FIELD?
 025650  001411                             BEQ     T16L08          ;BRANCH IF NO
 025652  076600                             MED                     ;GET TAG LOG INFO.
 025654  000107                             .WORD   RTAG
 025656  000300                             SWAB    R0              ;PUT TAG IN LOW BYTE
 025660  042700  177400                     BIC     #177400,R0      ;LOOK AT TAG ONLY
 025664  010037  001164                     MOV     R0,$REG3        ;SAVE BAD DATA
 025670  104070                             ERROR   70              ;ERROR: TEST OF MSB ADDR. (A10) TO ADDRESS FIELD FAILED
                                                                    ;       TAG PARITY ERROR
 025672  000417                             BR      T16L04          ;GO TO END OF TEST

 025674  032737  000100  177744     T16L08: BIT     #100,@#EREG     ;LOW BYTE P.E.?
 025702  001406                             BEQ     T16L09          ;BRANCH IF NO
 025704  076600                             MED                     ;GET LOG INFORMATION
 025706  000106                             .WORD   CDL
 025710  010037  001164                     MOV     R0,$REG3        ;SAVE INFORMATION
 025714  104071                             ERROR   71              ;ERROR: TEST OF MSB ADDR. (A10) TO ADDRESS FIELD FAILED
                                                                    ;       LOW BYTE PARITY ERROR
 025716  000405                             BR      T16L04          ;GO TO END OF TEST

 025720                             T16L09:
 025720  076600                             MED                     ;GET LOG INFORMATION
 025722  000106                             .WORD   CDH
 025724  010037  001164                     MOV     R0,$REG3        ;SAVE INFORMATION
 025730  104072                             ERROR   72              ;ERROR: TEST OF MSB ADDR. (A10) TO TAG FIELD FAILED
                                                                    ;       HIGH BYTE PARITY ERROR

 025732  005037  177572             T16L04: CLR     @#MMR0          ;KT OFF
 025736  012737  033142  000114             MOV     #UPERR,@#PVEC   ;RESTORE PARITY ERROR HANDLER
 025744  000401                             BR      TST32           ;;GO TO NEXT TEST

 025746  000000                     LAST1:  .WORD   0               ;TEST ADDRESS LOCATION

                                    ;;*****************************************************************
                                    ;*TEST 32        TEST OF MSB ADDRESS (A10) TO CACHE DATA FIELD
                                    ;*
                                    ;*   THIS TEST CHECKS FOR DUAL ADDRESSING ON THE DATA FIELD
                                    ;*FOR THE MSB (A10) ADDRESS TO CACHE.  THERE ARE TWO PASSES.
                                    ;*THE FIRST EXERCISES THE DATA BITS AND THE SECOND EXERCISES
                                    ;*THE DATA PARITY BITS.  THE TEST DATA IS STORED IN A TABLE
                                    ;*(TPAT) AT THE END OF THE TEST.  INITIALLY TEST ADDRESSES
                                    ;*ARE CALCULATED WHICH DON'T OVERLAP THE TEST INSTRUCTIONS
                                    ;*AND WHICH HAVE THE SAME CACHE ADDRESS (A1-A9) EXCEPT FOR
                                    ;*A10.  ONE ADDRESS IS THE LAST LOC IN THIS TEST (TAD1)
                                    ;*AND THE SECOND LIES IN A 1K BUFFER AT THE END OF THE
                                    ;*PROGRAM.  A SUBROUTINE, HAD, GENERATES THIS SECOND ADDRESS.
                                    ;*ON THE FIRST PASS DIFFERENT TEST DATA IS WRITTEN IN THE
                                    ;*CONGRUENT ADDRESS AND THEN CHECKED TO BE A HIT AND TO
                                    ;*BE THE CORRECT VALUE.  ON THE SECOND PASS NEW DATA IS
                                    ;*CHOSEN, WHICH GENERATES OPPOSITE PARITY IN THE DATA FIELD.
                                    ;*IS WRITTEN IN THE ADDRESSES AND THEN CHECKED TO BE A HIT
                                    ;*AND TO BE THE CORRECT VALUE.  ANY PARITY ERRORS OR HIT
                                    ;*ERRORS ARE REPORTED.
                                    ;*   R2 CONTAINS THE CONGRUENT ADDRESS FOR TAD1

                                    ;;*****************************************************************
 025750  012737  000214  177746     TST32:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
```

```
 025756  000004                             SCOPE
 025760  012737  026444  001234             MOV     #TST33,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 025766  012737  026224  000114             MOV     #T17L01,@#PVEC  ;SET UP FOR PARITY ERRORS
 025774  004737  033714                     JSR     PC,HAD          ;CALC CONGRUENT ADDRESS IN TEST BUFFER
 026000  026442                             .WORD   TAD1            ;TEST ADDRESS
 026002  013702  001172                     MOV     @TMP0,R2        ;SAVE CONGRUENT ADDRESS
 026006  005000                             CLR     R0              ;INIT TEST PATTERN ADDRESS REG
 026010  012737  000200  177746     T17L08: MOV     #200,@#CCR      ;ALL CACHE ON
 026016  016037  026432  026442     T17L07: MOV     TPAT(R0),@#TAD1 ;WRITE CACHE LOCS WITH
 026024  016012  026436                     MOV     TPAT+4(R0),(R2) ;ADDRESS BIT A10 COMPLEMENTED
 026030  013701  026442                     MOV     @#TAD1,R1       ;SEE IF DATA IN CACHE
 026034  033727  177752  000004             BIT     @#HMR,#HMR2     ;HIT?
 026042  001420                             BEQ     T17L02          ;BRANCH IF NO TO ERROR
 026044  020160  026432                     CMP     R1,TPAT(R0)     ;DATA CORRECT?
 026050  001051                             BNE     T17L03          ;BRANCH IF NO TO ERROR
 026052  011201                             MOV     (R2),R1         ;SEE IF NEXT DATA IN CACHE
 026054  033727  177752  000004             BIT     @#HMR,#HMR2     ;HIT?
 026062  001425                             BEQ     T17L04          ;BRANCH IF NO TO ERROR
 026064  020160  026436                     CMP     R1,TPAT+4(R0)   ;DATA OK?
 026070  001030                             BNE     T17L05          ;BRANCH IF NO TO ERROR
 026072  005760  026436                     TST     TPAT+4(R0)      ;TEST IF FIRST PASS
 026076  100151                             BPL     T17L06          ;BRANCH TO END OF TEST IF NO
 026100  005720                             TST     (R0)+           ;UPDATE ADDRESS
 026102  000745                             BR      T17L07          ;GO TEST NEW DATA

 026104  052737  000014  177746     T17L02: BIS     #14,@#CCR       ;CACHE OFF
 026112  012737  026442  001162             MOV     #TAD1,$REG2     ;SAVE ADDRESS
 026120  012737  026010  001110     T17L09: MOV     #T17L08,@#$LPERR ;INIT. RETURN FOR ERROR LOOP
 026126  005037  001160                     CLR     $REG1           ;SAVE ADDRESS
 026132  104062                             ERROR   62              ;ERROR: TEST OF MSB ADDRESS (A10) TO DATA FIELD FAILED
                                                                    ;       ADDRESS COULD NOT BE MADE A HIT
 026134  000532                             BR      T17L06          ;GO TO END OF TEST

 026136  052737  000014  177746     T17L04: BIS     #14,@#CCR       ;CACHE OFF
 026144  010237  001162                     MOV     R2,$REG2        ;SAVE ADDRESS
 026150  000763                             BR      T17L09          ;REPORT ERROR

 026152  052737  000014  177746     T17L05: BIS     #14,@#CCR       ;CACHE OFF
 026160  016037  026436  001166             MOV     TPAT+4(R0),$REG4 ;SAVE GOOD DATA
 026166  010237  001162                     MOV     R2,$REG2        ;SAVE BAD ADDRESS
 026172  000405                             BR      T17L10          ;REPORT ERROR

 026174  052737  000014  177746     T17L03: BIS     #14,@#CCR       ;CACHE OFF
 026202  016037  026432  001166             MOV     TPAT(R0),$REG4  ;SAVE GOOD DATA
 026210  010137  001164             T17L10: MOV     R1,$REG3        ;SAVE BAD DATA
 026214  005037  001160                     CLR     $REG1           ;SAVE BAD ADDRESS
 026220  104063                             ERROR   63              ;ERROR: TEST OF MSB ADDR. (A10) TO DATA FIELD FAILED
                                                                    ;       ADDRESS HELD WRONG DATA
 026222  000477                             BR      T17L06          ;GO TO END OF TEST

 026224  052737  000014  177746     T17L01: BIS     #14,@#CCR       ;CACHE OFF

 026232  010046                             MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
 026234  076600                             MED                     ;GET CONTENTS OF LOG REG
 026236  000022                             .WORD   RLOG
 026240  052700  100001                     BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
```

```
026244  076600                           MED                      ;UNLOCK ERROR LOG
026246  000222                           .WORD   WLOG
026250  012600                           MOV     (SP)+,R0         ;RESTORE R0

026252  011637  001164                   MOV     (SP),$REG3       ;GET PC+2 OF ERROR
026256  162737  000002  001164           SUB     #2,$REG3         ;SAVE PC OF ERROR
026264  022626                           CMP     (SP)+,(SP)+      ;RESTORE STACK
026266  076600                           MED                      ;GET LOG INFOR FOR PHY. ADDR. A17,A16
026270  000101                           .WORD   RSER
026272  000300                           SWAB    R0               ;PUT PHY. ADDR A17, A16 IN LOW BYTE
026274  042700  177776                   BIC     #177776,R0       ;ONLY LOOK AT A17, A16
026300  010037  001160                   MOV     R0,$REG1         ;SAVE ADDRESS
026304  076600                           MED                      ;GET LOG INFORMATION
026306  000102                           .WORD   LOADD
026310  010037  001162                   MOV     R0,$REG2         ;SAVE INFORMATION
026314  076600                           MED                      ;GET LOG INFORMATION
026316  000100                           .WORD   RJAM
026320  032700  000400                   BIT     #400,R0          ;ERROR IN BACKING STORE
026324  001402                           BEQ     T17L11           ;BRANCH IF NO
026326  104001                           ERROR   1                ;ERROR: UNEXP. PARITY ERROR IN BACKING STORE
026330  000434                           BR      T17L06           ;GO TO END OF TEST

026332  032737  000100  177744  T17L11:  BIT     #100,@#EREG      ;PARITY ERROR LOW BYTE?
026340  001406                           BEQ     T17L12           ;BRANCH IF NO
026342  076600                           MED                      ;GET LOG INFORMATION
026344  000106                           .WORD   CDL
026346  010037  001164                   MOV     R0,$REG3         ;SAVE INFORMATION
026352  104064                           ERROR   64               ;ERROR: TEST OF MSB  ADDR. (A10) TO DATA FIELD FAILED
                                                                  ;        PARITY ERROR LOW BYTE
026354  000422                           BR      T17L06           ;GO TO END OF TEST

026356  032737  000200  177744  T17L12:  BIT     #200,@#EREG      ;PARITY ERROR HIGH BYTE?
026364  001406                           BEQ     T17L13           ;BRANCH IF NO
026366  076600                           MED                      ;GET LOG INFORMATION
026370  000106                           .WORD   CDL
026372  010037  001164                   MOV     R0,$REG3         ;SAVE INFORMATION
026376  104065                           ERROR   65               ;ERROR: TEST OF MSB ADDR. (A10) TO DATA FIELD FAILED
                                                                  ;        PARITY ERROR HIGH BYTE
026400  000410                           BR      T17L06           ;GO TO END OF TEST

026402                          T17L13:
026402  076600                           MED                      ;GET TAG LOG INFO.
026404  000107                           .WORD   RTAG
026406  000300                           SWAB    R0               ;PUT TAG IN LOW BYTE
026410  042700  177400                   BIC     #177400,R0       ;LOOK AT TAG ONLY
026414  010037  001164                   MOV     R0,$REG3         ;SAVE BAD DATA
026420  104066                           ERROR   66               ;ERROR: TEST OF MSB ADDR. (A10) TO DATA FIELD FAILED
                                                                  ;        PARITY ERROR TAG


026422  012737  033142  000114  T17L06:  MOV     #UPERR,@#PVEC    ;RESTORE PARITY ERROR HANDLER
026430  000405                           BR      TST33            ;;GO TO NEXT TEST


026432  066666                  TPAT:    .WORD   66666            ;TEST DATA FOR DATA BIT TEST
026434  000401                           .WORD   401              ;TEST DATA FOR PARITY BIT TEST
026436  111111                           .WORD   111111           ;TEST DATA FOR DATA BIT TEST
```

```
026440  001403                           .WORD   1403             ;TEST DATA FOR PARITY BIT TEST

026442  000000                  TAD1:    .WORD   0       ;TEST ADDRESS

                                ;;*************************************************************
                                ;*TEST 33       TEST CACHE IS NOT ALLOCATED DURING ODD ADDRESS TRAP
                                ;*
                                ;*THIS TEST FIRST PUTS DATA IN A CACHE LOC.  THEN A WORD
                                ;*INSTRUCTION TO AN ODD BYTE ADDRESS TRIES TO CLEAR THE
                                ;*LOC AND FORCE AN ODD ADDRESS ERROR.  UPON TRAPPING, THE
                                ;*LOC IN CACHE IS LOOKED AT AND VERIFIED TO NOT HAVE CHANGED.

                                ;;*************************************************************
026444  012737  000214  177746  TST33:   MOV     #214,@#CCR       ;CACHE OFF FOR SCOPE
026452  000004                           SCOPE
026454  012737  026634  001234           MOV     #TST34,SKTST     ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
026462  012737  000200  177746           MOV     #200,@#CCR       ;CACHE ON
026470  012737  026510  000004           MOV     #T27L01,@#4      ;SETUP FOR ODD ADDRESS TRAP
026476  012737  177777  060000           MOV     #177777,@#BUFL   ;PUT DATA IN CACHE
026504  005037  060001                   CLR     @#BUFL+1         ;FORCE ODD ADDRESS ERROR

026510  022626                  T27L01:  CMP     (SP)+,(SP)+      ;RESTORE THE STACK

026512  010046                           MOV     R0,-(SP)         ;SAVE R0 FOR MED INST
026514  076600                           MED                      ;GET CONTENTS OF LOG REG
026516  000022                           .WORD   RLOG
026520  052700  100001                   BIS     #100001,R0       ;ENABLE ERROR LOG & LOG FIRST MODE
026524  076600                           MED                      ;UNLOCK ERROR LOG
026526  000222                           .WORD   WLOG
026530  012600                           MOV     (SP)+,R0         ;RESTORE R0

026532  013700  060000                   MOV     @#BUFL,R0        ;GET DATA
026536  033727  177752  000004           BIT     @#HMR,#HMR2      ;HIT?
026544  001407                           BEQ     T27L02           ;BRANCH TO ERROR IF NO
026546  020027  177777                   CMP     R0,#177777       ;DATA UNCHANGED?
026552  001016                           BNE     T27L03           ;BRANCH IF YES TO ERROR
026554  012737  033352  000004  T27L04:  MOV     #UT4,@#4         ;RESTORE HANDLER FOR UNEXPECTED TRAPS TO 4
026562  000424                           BR      TST34            ;;GO TO NEXT TEST

026564  052737  000014  177746  T27L02:  BIS     #14,@#CCR        ;CACHE OFF
026572  005037  001160                   CLR     $REG1            ;SAVE FAILING ADDRESS
026576  012737  060000  001162           MOV     #BUFL,$REG2      ;SAVE FAILING ADDRESS
026604  104043                           ERROR   43               ;ERROR: ADDRESS COULD NOT BE MADE A HIT
026606  000762                           BR      T27L04           ;GO TO END OF TEST

026610  032737  000014  177746  T27L03:  BIT     #14,@#CCR        ;CACHE OFF
026616  005037  001160                   CLR     $REG1            ;SAVE BAD ADDRESS
026622  012737  060001  001162           MOV     #BUFL+1,$REG2    ;SAVE BAD ADDRESS
026630  104116                           ERROR   116              ;ERROR: CACHE ALLOCATED DURING ODD ADDRESS TRAP
026632  000752                           BR      T27L04           ;GO TO END OF TEST

                                ;;*************************************************************
                                ;*TEST 34       TEST CACHE NOT ALLOCATED DURING RED ZONE TRAP
                                ;*
                                ;*   THIS TEST FIRST PUTS DATA IN A CACHE LOC WHICH COR-
                                ;*RESPONDS TO A RED ZONE ADDRESS.  A STACK OPERATION IS
```

```
                                          ;*DONE TO THIS ADDRESS WHICH WILL CHANGE THE DATA IF
                                          ;*COMPLETED.  UPON TRAPPING, THE DATA IN CACHE IS LOOKED
                                          ;*AT AND VERIFIED TO NOT HAVE CHANGED.

                                          ;;*****************************************************************
 026634  012737  000214  177746   TST34:  MOV     #214,@#CCR        ;CACHE OFF FOR SCOPE
 026642  000004                           SCOPE
 026644  012737  027030  001234           MOV     #TST35,SKTST      ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 026652  012737  000200  177746           MOV     #200,@#CCR        ;CACHE ON
 026660  012737  026706  000004           MOV     #T28L01,@#4       ;SET UP FOR RED ZONE TRAPS
 026666  005037  177774                   CLR     @#177774          ;INITIALIZE THE STACK LIMIT REG
 026672  005037  000336                   CLR     @#336             ;INITIALIZE TEST LOC
 026676  012706  000336                   MOV     #336,SP           ;PUT RED ZONE TRAP ADDRESS IN STACK PTER
 026702  012716  177777                   MOV     #177777,(SP)      ;FORCE RED ZONE TRAP

 026706  012706  001100           T28L01: MOV     #1100,SP          ;RESTORE THE STACK

 026712  010046                           MOV     R0,-(SP)          ;SAVE R0 FOR MED INST
 026714  076600                           MED                       ;GET CONTENTS OF LOG REG
 026716  000022                           .WORD   RLOG
 026720  052700  100001                   BIS     #100001,R0        ;ENABLE ERROR LOG & LOG FIRST MODE
 026724  076600                           MED                       ;UNLOCK ERROR LOG
 026726  000222                           .WORD   WLOG
 026730  012600                           MOV     (SP)+,R0          ;RESTORE R0

 026732  013700  000336                   MOV     @#336,R0          ;GET DATA
 026736  033727  177752  000004           BIT     @#HMR,#HMR2       ;HIT?
 026744  001412                           BEQ     T28L02            ;BRANCH IF NO
 026746  005700                           TST     R0                ;DATA UNCHANGED?
 026750  001022                           BNE     T28L03            ;BRANCH IF NO TO ERROR
 026752  012737  033352  000004   T28L04: MOV     #UT4,@#4          ;RESTORE HANDLER FOR UNEXP. TRAPS TO 4
 026760  005037  000000                   CLR     @#0               ;RESTORE LOC 0
 026764  005037  000002                   CLR     @#2               ;RESTORE LOC 2
 026770  000417                           BR      TST35             ;;GO TO NEXT TEST

 026772  052737  000014  177746   T28L02: BIS     #14,@#CCR         ;CACHE OFF
 027000  005037  001160                   CLR     $REG1             ;SAVE FAILING ADDR.
 027004  012737  000336  001162           MOV     #336,$REG2        ;SAVE FAILING ADDR.
 027012  104043                           ERROR   43                ;ERROR: ADDRESS COULD NOT BE MADE A HIT
 027014  000756                           BR      T28L04            ;GO TO END OF TEST

 027016  052737  000014  177746   T28L03: BIS     #14,@#CCR         ;CACHE OFF
 027024  104117                           ERROR   117               ;ERROR: CACHE ALLOCATED DURING RED ZONE TRAP
 027026  000751                           BR      T28L04            ;GO TO END OF TEST

                                          ;;*****************************************************************
                                          ;*TEST 35        TEST CACHE NOT ALLOCATED DURING KT ABORT
                                          ;*
                                          ;*   DATA IS PUT IN CACHE IN A TEST BUFFER ADDRESS.  KIPAR4
                                          ;*IS SET UP TO  REFERENCE THAT ADDRESS AND KIPDR4 IS SET
                                          ;*UP TO ABORT ACCESSES TO NON RESIDENT PAGE.  THE KT IS
                                          ;*TURNED ON AND A MEMORY REFERENCE THROUGH KIPAR4 IS MADE
                                          ;*WHICH WOULD MODIFY THE TEST LOCATION IF COMPLETED.  UPON
                                          ;*TRAPPING, THE LOCATION IS LOOKED AT AND VERIFIED TO NOT
                                          ;*HAVE CHANGED.
                                          ;*   IF THE INHIBIT TEST USING KT SWITCH (SW12) IS SET,
```

```
                                          ;*THIS TEST IS SKIPPED.


                                          ;;*****************************************************************
 027030  012737  000014  177746   TST35:  MOV     #14,@#CCR         ;CACHE OFF FOR SCOPE
 027036  000004                           SCOPE
 027040  012737  027300  001234           MOV     #TST36,SKTST      ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 027046  032777  010000  152060           BIT     #SW12,@SWR        ;INHIBIT TESTS USING KT?
 027054  001111                           BNE     TST36             ;;YES, GO TO NEXT TEST
 027056  052737  000200  036034           BIS     #200,@#$KT11      ;USE KT FOR $SIZE
 027064  004737  035750                   JSR     PC,$SIZE          ;USE $SIZE TO SET UP PAR'S AND PDR'S
 027070  012737  027152  000250           MOV     #T29L01,@#250     ;SET UP FOR KT ABORTS
 027076  012737  077400  172310           MOV     #77400,KIPDR4     ;SET UP PDR4 TO ABORT ACCESS TO NON RESIDENT PAGE
 027104  013700  060000                   MOV     $BUFL,R0          ;GET TEST ADDRESS
 027110  042700  160000                   BIC     #160000,R0        ;MASK ITS PAR ADDRESS
 027114  052700  100000                   BIS     #100000,R0        ;HAVE IT ADDRESS PAR4
 027120  013737  172346  172350           MOV     @#KIPAR3,@#KIPAR4  ;INIT PAR4 TO HAVE SAME OFFSET AS PAR3 FOR THE BUFFER
 027126  012737  000200  177746           MOV     #200,@#CCR        ;CACHE ON
 027134  012737  177777  060000           MOV     #177777,@#BUFL    ;INIT TEST ADDRESS
 027142  052737  000001  177572           BIS     #1,@#MMR0         ;KT ON
 027150  005010                           CLR     (R0)              ;FORCE KT ABORT

 027152  022626                   T29L01: CMP     (SP)+,(SP)+       ;RESTORE STACK

 027154  010046                           MOV     R0,-(SP)          ;SAVE R0 FOR MED INST
 027156  076600                           MED                       ;GET CONTENTS OF LOG REG
 027160  000022                           .WORD   RLOG
 027162  052700  100001                   BIS     #100001,R0        ;ENABLE ERROR LOG & LOG FIRST MODE
 027166  076600                           MED                       ;UNLOCK ERROR LOG
 027170  000222                           .WORD   WLOG
 027172  012600                           MOV     (SP)+,R0          ;RESTORE R0

 027174  013701  060000                   MOV     @#BUFL,R1         ;GET ADDRESS
 027200  033727  177752  000004           BIT     @#HMR,#HMR2       ;HIT?
 027206  001415                           BEQ     T29L02            ;BRANCH IF NO
 027210  020127  177777                   CMP     R1,#177777        ;DATA OK?
 027214  001024                           BNE     T29L03            ;BRANCH IF NO
 027216  042737  000001  177572   T29L04: BIC     #1,@#MMR0         ;KT OFF
 027224  052737  000006  172310           BIS     #6,@#KIPDR4       ;ALLOW READ OR WRITE TO PAGE
 027232  012737  000252  000250           MOV     #252,@#250        ;RESTORE KT TRAP CATCHER
 027240  000417                           BR      TST36             ;;GO TO NEXT TEST

 027242  052737  000014  177746   T29L02: BIS     #14,@#CCR         ;CACHE OFF
 027250  005037  001160                   CLR     $REG1             ;SAVE FAILING ADDRESS
 027254  012737  060000  001160           MOV     #BUFL,$REG1       ;SAVE FAILING ADDRESS
 027262  104043                           ERROR   43                ;ERROR: ADDRESS COULD NOT BE MADE A HIT
 027264  000754                           BR      T29L04            ;GO TO END OF TEST

 027266  052737  000014  177746   T29L03: BIS     #14,@#CCR         ;CACHE OFF
 027274  104120                           ERROR   120               ;ERROR: CACHE ALLOCATED DURING KT ABORT
 027276  000747                           BR      T29L04            ;GO TO END OF TEST

                                          ;;*****************************************************************
                                          ;*TEST 36        DYNAMIC TEST OF CACHE
                                          ;*
                                          ;*   THIS TEST CREATES A GREAT DEAL OF ACTIVITY IN CACHE
                                          ;*TO TRY TO FIND ANY NOISE OR TIMING PROBLEMS.  THESE
```

```
                                           ;*PROBLEMS WILL BE DETECTED VIA THE PARITY ERRORS, ILLEGAL
                                           ;*INSTRUCTION TRAPS OR DATA CHANGES THEY CAUSE.  FIRST
                                           ;*CACHE IS LOADED WITH AN ALTERNATING DATA PATTERN (525,252).
                                           ;*THEN IT IS REFERENCED AS QUICKLY AS POSSIBLE IN OPPOSITE
                                           ;*DIRECTIONS TO CAUSE LARGE CHANGES IN THE ADDRESS LINES AND
                                           ;*RAPID CHANGES IN THE DATA LINES.  THIS IS THEN REPEATED
                                           ;*WITH A DIFFERENT DATA PATTERN AND THE CACHE IS MODIFIED
                                           ;*AS THE REFERENCES OCCUR.  AFTER THIS THE LOCATIONS ARE
                                           ;*CHECKED TO CONTAIN THEIR PROPER VALUES.
                                           ;*   FOLLOWING THIS, THE TAG FIELD IS WRITTEN WITH A
                                           ;*CHANGING PATTERN.  THEN THE CACHE IS REFERENCED AS QUICKLY
                                           ;*AS POSSIBLE IN OPPOSITE DIRECTIONS TO CAUSE LARGE CHANGES
                                           ;*IN THE ADDRESS LINES AND RAPID CHANGES IN THE TAG FIELD.
                                           ;*THIS LAST PART IS SKIPPED IF THE INHIBIT TEST USING KT
                                           ;*SWITCH (SW12) IS SET.

                                           ;;*****************************************************************
 027300  012737  000214  177746  TST36:  MOV     #214,@#CCR         ;CACHE OFF FOR SCOPE
 027306  000004                          SCOPE
 027310  012737  030260  001234          MOV     #TST37,$KTST       ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
 027316  012737  030036  000114          MOV     #T18L01,@#PVEC     ;SETUP FOR PARITY ERRORS
 027324  012737  027710  000010          MOV     #T18L02,@#10       ;SETUP FOR TRAPS TO ILLEGAL INST
 027332  012737  027340  001110          MOV     #T18L11,@#$LPERR   ;INIT RETURN FOR ERROR LOOPS
 027340  012737  000200  177746  T18L11: MOV     #200,@#CCR         ;CACHE ON

                                         ;GENERATE TEST DATA IN A 1K BUFFER

 027346  012703  060000                  MOV     #BUFL,R3           ;GET STARTING ADDRESS OF BUFFER
 027352  012702  002000                  MOV     #2000,R2           ;INIT REG FOR 1K COUNT
 027356  012701  176540                  MOV     #176540,R1         ;PUT RANDOM # IN REG
 027362  012700  023456                  MOV     #023456,R0         ;PUT RANDOM # IN REG
 027366  060001                  1$:     ADD     R0,R1              ;GENERATE NEW RANDOM DATA
 027370  010123                          MOV     R1,(R3)+           ;SAVE DATA
 027372  000261                          SEC                        ;GENERATE MORE
 027374  006101                          ROL     R1                 ;RANDOM DATA
 027376  006000                          ROR     R0
 027400  077206                          SOB     R2,1$              ;LOOP TILL 1K BUFFER FULL

                                         ;LOAD CACHE WITH PATTERN AND TEST CACHE

 027402  012700  060000                  MOV     #BUFL,R0           ;SET UP TO ADDRESS BUFFER
 027406  012701  060000                  MOV     #BUFL,R1           ;ASET UP TO ADDRESS BUFFER
 027412  012702  002000                  MOV     #2000,R2           ;INIT REG FOR 1K COUNT
 027416  012703  002000                  MOV     #2000,R3           ;INIT REG FOR 1K COUNT
 027422  005721                  2$:     TST     (R1)+              ;GET DATA IN CACHE
 027424  077202                          SOB     R2,2$              ;LOOP TILL 1K REFERENCED
 027426  022041                  3$:     CMP     (R0)+,-(R1)        ;REFERENCE CACHE QUICKLY AND WITH COMPLEMENT ADDR
 027430  077302                          SOB     R3,3$              ;LOOP TILL ALL CACHE REFERENCED

                                         ;GENERATE SECOND TEST PATTERN IN BUFFER AND TEST IT

 027432  012700  060000                  MOV     #BUFL,R0           ;SET UP TO ADDRESS BUFFER
 027436  012701  060000                  MOV     #BUFL,R1           ;SET UP TO ADDRESS BUFFER
 027442  012702  002000                  MOV     #2000,R2           ;INIT REG FOR 1K COUNT
 027446  012703  002000                  MOV     #2000,R3           ;INIT REG FOR 1K COUNT
 027452  005004                          CLR     R4                 ;INIT DATA
```

```
 027454  010421                  5$:     MOV     R4,(R1)+           ;LOAD BUFFER WITH PATTERN
 027456  005204                          INC     R4                 ;CHANGE DATA
 027460  077203                          SOB     R2,5$              ;LOOP TILL 1K LOADED

 027462  062041                  6$:     ADD     (R0)+,-(R1)        ;REFERENCE CACHE QUICKLY
 027464  077302                          SOB     R3,6$              ;LOOP TILL ALL CACHE REFERENCED

                                         ;CHECK DATA IN CACHE OR MAIN MEM CORRECT

 027466  012701  001777                  MOV     #1777,R1           ;INIT REG WITH GOOD DATA
 027472  012702  001000                  MOV     #1000,R2           ;INIT REG FOR 1/2K COUNT
 027476  014003                  7$:     MOV     -(R0),R3           ;GET DATA
 027500  020103                          CMP     R1,R3              ;DATA OK?
 027502  001140                          BNE     T18L03             ;BRANCH IF NO TO ERROR
 027504  077204                          SOB     R2,7$              ;LOOP TILL 1/2K REFERRED
 027506  012702  001000                  MOV     #1000,R2           ;INIT REG FOR 1/2K COUNT
 027512  012701  002776                  MOV     #2776,R1           ;INIT REG WITH 'GOOD' DATA
 027516  014003                  10$:    MOV     -(R0),R3           ;GET DATA
 027520  020103                          CMP     R1,R3              ;DATA OK?
 027522  001130                          BNE     T18L03             ;BRANCH IF NO TO ERROR
 027524  005301                          DEC     R1                 ;ADJUST GOOD DATA
 027526  077205                          SOB     R2,10$             ;LOOP TILL ALL DATA CHECKED

                                         ;NOW TEST TAG MEM

 027530  032777  010000  151376          BIT     #SW12,@SWR         ;INHIBIT TESTS USING KT?
 027536  001402                          BEQ     11$                ;CONTINUE TEST IF NO
 027544  000137  030260                  JMP     @#TST37            ;GO TO NEXT TEST
 027544  052737  000200  036034  11$:    BIS     #200,@#$KT11       ;KT ON FOR $SIZE
 027552  004737  035750                  JSR     PC,@SIZE           ;SIZE MEMORY
 027556  013737  027564  001110          MOV     T18L05,@#$LPERR    ;INIT RETURN FOR ERROR LOOPS
 027564  012737  000200  177746  T18L05: MOV     #200,@#CCR         ;CACHE ON
 027572  012700  100000                  MOV     #100000,R0         ;HAVE R0 ADDRESS PAR4
 027576  012701  120000                  MOV     #120000,R1         ;HAVE R1 ADDR. PAR5
 027602  012704  172350                  MOV     #KIPAR4,R4         ;PUT PAR4 ADDR IN R4
 027606  012705  172352                  MOV     #KIPAR5,R5         ;PUT PAR5 ADDR IN R5
 027612  013702  036322                  MOV     @#$LSTBK,R2        ;GET LAST BANK
 027616  010215                          MOV     R2,(R5)            ;SET UP PAR5
 027620  010214                  T18L06: MOV     R2,(R4)            ;SET UP PAR4
 027622  052737  000001  177572          BIS     #1,@#MMR0          ;KT ON
 027630  005720                  T18L07: TST     (R0)+              ;WRITE CACHE VIA DATI
 027632  032700  003776                  BIT     #3776,R0           ;ALL CACHE WRITTEN?
 027636  001404                          BEQ     T18L09             ;BRANCH IF YES
 027640  162714  000040                  SUB     #40,(R4)           ;CALC NEW PAR4 TO GIVE NEW TAG PATTERN
 027644  100371                          BPL     T18L07             ;WRITE CACHE IF TAG > OR EQUAL TO 0
 027646  000764                          BR      T18L06             ;GO INIT PAR4 TO RESTART PATTERN

 027650  022140                  T18L09: CMP     (R1)+,-(R0)        ;REFERENCE CACHE
 027652  032701  003776                  BIT     #3776,R1           ;ALL CACHE TESTED?
 027656  001002                          BNE     2$                 ;BRANCH IF NO TO CONTINUE
 027660  000137  030234                  JMP     T18L10             ;GO TO END OF TEST
 027664  162715  000040          2$:     SUB     #40,(R5)           ;ADJUST PAR5 FOR NEXT TEST ADDR. REF.
 027670  100001                          BPL     1$                 ;TAG > OR EQUAL 0, BRANCH IF YES
 027672  010215                          MOV     R2,(R5)            ;NO, INIT PAR5 FOR HIGHEST TAG ADDR
 027674  062714  000040          1$:     ADD     #40,(R4)           ;ADJUST PAR4 FOR NEXT TEST ADDR.
 027700  020214                          CMP     R2,(R4)            ;IS PAR4 > MAX ADDRESS?
```

```
027702  002362                          BGE     T10L09          ;GO TEST IT IF NO
027704  005014                          CLR     (R4)            ;RESTART PAR4 AT LOW TEST ADDR
027706  000760                          BR      T10L09          ;GO TEST IT

027710  052737  000014  177746  T10L02: BIS     #14,@#CCR       ;CACHE OFF
027716  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 OF TRAP
027722  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF TRAP
027730  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
027732  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
027734  000101                          .WORD   RSER
027736  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
027740  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
027744  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
027750  076600                          MED                     ;GET LOG INFORMATION
027752  000102                          .WORD   LOADD
027754  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION

027760  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
027762  076600                          MED                     ;GET CONTENTS OF LOG REG
027764  000022                          .WORD   RLOG
027766  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
027772  076600                          MED                     ;UNLOCK ERROR LOG
027774  000222                          .WORD   WLOG
027776  012600                          MOV     (SP)+,R0        ;RESTORE R0

030000  104074                          ERROR   74              ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                                                ;       TRAP TO 10 OCCURRED
030002  000514                          BR      T10L10          ;GO TO END OF TEST

030004  052737  000014  177746  T10L03: BIS     #14,@#CCR       ;CACHE OFF
030012  005037  001160                  CLR     $REG1           ;SAVE ADDRESS
030016  010037  001162                  MOV     R0,$REG2        ;SAVE ADDRESS
030022  010337  001164                  MOV     R3,$REG3        ;SAVE BAD DATA
030026  010137  001166                  MOV     R1,$REG4        ;SAVE GOOD DATA
030032  104073                          ERROR   73              ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                                                ;       LOC HELD WRONG DATA
030034  000477                          BR      T10L10          ;GO TO NEXT TEST

030036  052737  000014  177746  T10L01: BIS     #14,@#CCR       ;CACHE OFF

030044  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
030046  076600                          MED                     ;GET CONTENTS OF LOG REG
030050  000022                          .WORD   RLOG
030052  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
030056  076600                          MED                     ;UNLOCK ERROR LOG
030060  000222                          .WORD   WLOG
030062  012600                          MOV     (SP)+,R0        ;RESTORE R0

030064  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 OF TRAP
030070  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF TRAP
030076  022626                          CMP     (SP)+,(SP)+     ;ADJUST STACK
030100  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
030102  000101                          .WORD   RSER
030104  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
030106  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
030112  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
```

```
030116  076600                          MED                     ;GET LOG INFORMATION
030120  000102                          .WORD   LOADD
030122  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
030126  076600                          MED                     ;GET LOG INFORMATION
030130  000100                          .WORD   RJAN
030132  032700  000400                  BIT     #400,R0         ;ERROR IN BACKING STORE?
030136  001402                          BEQ     T10L12          ;BRANCH IF NO
030140  104001                          ERROR   1               ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE
030142  000434                          BR      T10L10          ;GO TO NEXT TEST

030144  032737  000100  177744  T10L12: BIT     #100,@#EREG     ;LOW BYTE PE?
030152  001406                          BEQ     T10L13          ;BRANCH IF NO
030154  076600                          MED                     ;GET LOG INFORMATION
030156  000106                          .WORD   CDL
030160  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
030164  104075                          ERROR   75              ;ERROR: DYNAMIC TEST OF CACHE FAILED

                                                                ;       LOW BYTE PARITY ERROR
030166  000422                          BR      T10L10          ;GO TO END OF TEST

030170  032737  000200  177744  T10L13: BIT     #200,@#EREG     ;HIGH BYTE PE?
030176  001406                          BEQ     T10L14          ;BRANCH IF NO
030200  076600                          MED                     ;GET LOG INFORMATION
030202  000105                          .WORD   CDH
030204  010037  001164                  MOV     R0,$REG3        ;SAVE INFORMATION
030210  104076                          ERROR   76              ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                                                ;       HIGH BYTE PARITY ERROR
030212  000410                          BR      T10L10          ;GO TO END OF TEST

030214                          T10L14:
030214  076600                          MED                     ;GET TAG LOG INFO.
030216  000107                          .WORD   RTAG
030220  000300                          SWAB    R0              ;PUT TAG IN LOW BYTE
030222  042700  177400                  BIC     #177400,R0      ;LOOK AT TAG ONLY
030226  010037  001164                  MOV     R0,$REG3        ;SAVE BAD DATA
030232  104077                          ERROR   77              ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                                                ;       TAG PARITY ERROR

030234  005037  177572          T10L10: CLR     @#MR0           ;KT OFF
030240  012737  000012  000010          MOV     #12,@#10        ;RESTORE TRAP CATCHER
030246  005037  000012                  CLR     @#12            ;RESTORE TRAP CATCHER
030252  012737  033142  000114          MOV     #UPERR,@#PVEC   ;RESTORE HANDLER FOR PARITY ERRORS


                                ;;*************************************************************
                                ;*TEST 37        TEST RETRIES TO BACKING STORE DONE ON CACHE PARITY TRAP
                                ;*
                                ;*   THE JAMUPP ON CACHE PARITY ERROR BIT IS CLEARED AND
                                ;*THE CACHE CONTROL REG IS TESTED TO CONTAIN THE CORRECT
                                ;*VALUE.  A CACHE LOC IS THEN WRITTEN WITH WRONG PARITY
                                ;*AND A TRAP IS FORCED.  THE LOC IS THEN REFERENCED TO SEE
                                ;*IF IT STILL IS IN CACHE (RETRY DONE).

                                ;;*************************************************************
030260  012737  000214  177746  TST37:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
030266  000004                          SCOPE
030270  012737  030524  001234          MOV     #TST40,SKTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
```

```
030276  012737  030350  000114          MOV     #1$,@#PVEC      ;SET UP FOR PARITY TRAP
030304  042737  000200  177746          BIC     #200,@#CCR      ;ENABLE RETRIES
030312  032737  000200  177746          BIT     #200,@#CCR      ;WAS BIT CLEARED?
030320  001045                          BNE     2$              ;BRANCH IF NO TO ERROR
030322  012737  000100  177746          MOV     #100,@#CCR      ;CACHE ON, WRITE WRONG PARITY, DO RETRIES
030330  005037  060000                  CLR     @#BUFL          ;WRITE WRONG PARITY
030334  012737  000000  177746          MOV     #0,@#CCR        ;WWP OFF
030342  005737  060000                  TST     @#BUFL          ;FORCE TRAP
030346  000445                          BR      3$              ;REPORT ERROR IF NO TRAP

030350  062706  000004          1$:     ADD     #4,SP           ;RESTORE THE STACK

030354  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
030356  076600                          MED                     ;GET CONTENTS OF LOG REG
030360  000022                          .WORD   RLOG
030362  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
030366  076600                          MED                     ;UNLOCK ERROR LOG
030370  000222                          .WORD   WLOG
030372  012600                          MOV     (SP)+,R0        ;RESTORE R0

030374  005737  060000                  TST     @#BUFL          ;SEE IF DATA IN CACHE
030400  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT?
030406  001036                          BNE     T23L01          ;GO TO END OF TEST IF YES
030410  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF

                                        ;RID CACHE OF BAD PARITY
030416  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
030424  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE


030430  104110                          ERROR   110             ;ERROR: RETRY TO BACKING STORE NOT DONE ON CACHE PARITY
030432  000424                          BR      T23L01          ;GO TO END OF TEST

030434  013737  177746  001160  2$:     MOV     @#CCR,$REG1     ;SAVE BAD DATA
030442  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF
030450  012737  000014  001162          MOV     #14,$REG2       ;SAVE GOOD DATA
030456  104026                          ERROR   26              ;ERROR: CACHE CONTROL REG HELD WRONG DATA
030460  000411                          BR      T23L01          ;GO TO END OF TEST

030462  012737  000214  177746  3$:     MOV     #214,@#CCR      ;CACHE OFF
030470  005037  001160                  CLR     $REG1           ;SAVE ADDR. OF TESTED LOC
030474  012737  060000  001162          MOV     #BUFL,$REG2     ;SAVE ADDR. OF TESTED LOC
030502  104042                          ERROR   42              ;ERROR: NO PARITY TRAP FROM LOC WRITTEN WITH WRONG PARIT

030504                          T23L01:

                                        ;RID CACHE OF BAD PARITY
030504  012737  000214  177746          MOV     #214,@#CCR      ;CACHE OFF IF ON
030512  004737  035134                  JSR     PC,SWEEP        ;GO PURGE CACHE


030516  012737  033142  000114          MOV     #UPERR,@#PVEC   ;RESTORE PARITY ERROR HANDLER

                                ;;*****************************************************************
                                ;*TEST 40       TEST DATO TO I/0 LOC NOT WRITTEN IN CACHE AND I/0
                                ;*
```

```
                                ;*   THE TEST INSTRUCTION ADDRESSES ARE FIRST EXAMINED TO
                                ;*DETERMINE IF THEY OVERLAP THE TEST LOCATION ADDRESS IN
                                ;*CACHE.  IF THEY DO, THE TEST IS RUN IN A NON OVERLAPPING
                                ;*ADDRESS SPACE.  A LOC IS PUT IN CACHE WHICH HAS THE SAME
                                ;*11 LEAST SIGNIFICANT ADDRESS BITS AS THE MEMORY MANAGEMENT
                                ;*REG KIPAR0.  A DATO IS THEN DONE TO KIPAR0 AND THE LOC
                                ;*IS CHECKED TO STILL BE IN CACHE.

                                ;;*****************************************************************
030524  012737  000214  177746  TST40:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
030532  000004                          SCOPE
030534  012737  030674  001234          MOV     #TST41,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
030542  012737  000200  177746          MOV     #200,@#CCR      ;TURN ON ALL OF CACHE
030550  012737  030610  001172          MOV     #T19L01,@#$TMP0 ;SAVE ADDRESS OF TEST INSTRUCTION
030556  042737  174000  001172          BIC     #174000,@#$TMP0 ;LOOK AT ITS CACHE ADDRESS
030564  023727  001172  002326          CMP     @#$TMP0,#2326   ;INSTRUCTION AT TEST LOC?
030572  002404                          BLT     T19L02          ;BRANCH IF NO
030574  023727  001172  002340          CMP     @#$TMP0,#2340   ;INSTRUCTION AT TEST LOC?
030602  003422                          BLE     T19L03          ;BRANCH IF YES
030604  005737  002340          T19L02: TST     @#2340          ;PUT TEST LOC IN CACHE
030610  005037  172340          T19L01: CLR     @#KIPAR0        ;DO DATO TO I/O
030614  005737  002340                  TST     @#2340          ;DATA STILL IN CACHE
030620  033727  177752  000004          BIT     @#HMR,#HMR2     ;WAS IT A HIT?
030626  001022                          BNE     TST41           ;;GO TO NEXT TEST IF YES
030630  012737  000003  001160  T19L04: MOV     #3,$REG1        ;SAVE PHYSICAL ADDRESS HIGH
030636  012737  172340  001162          MOV     #172340,$REG2   ;SAVE PHYSICAL ADDRESS LOW
030644  104025                          ERROR   25              ;ERROR: DATO TO I/O ADDRESS WRITTEN IN CACHE
030646  000412                          BR      TST41           ;;GO TO NEXT TEST

030650  005737  002340          T19L03: TST     @#2340          ;PUT TEST LOC IN CACHE
030654  005037  172340                  CLR     @#KIPAR0        ;DO DATO TO I/O
030660  005737  002340                  TST     @#2340          ;DATA STILL IN CACHE?
030664  033727  177752  000004          BIT     @#HMR,#HMR2     ;STILL A HIT?
030672  001756                          BEQ     T19L04          ;BRANCH TO ERROR IF NO

                                ;;*****************************************************************
                                ;*TEST 41       TEST CONSOLE INITIATED SWEEP INVALIDATES ALL CACHE
                                ;*
                                ;*   A LOC IS PUT IN CACHE, CHECKED TO BE A HIT AND THEN
                                ;*A CONSOLE SWEEP IS INITIATED.  THE LOC IS AGAIN REF-
                                ;*ERENCED TO SEE IF IT WAS INVALIDATED (NOT A HIT).  THIS
                                ;*IS DONE FOR ALL OF CACHE.  BEFORE THE CONSOLE SWEEP IS
                                ;*STARTED, THE TEST LOC IS VERIFIED TO NOT OVERLAP THE
                                ;*PROGRAM INSTRUCTION ADDRESSES IN CACHE.  IF THEY DO, THE
                                ;*TEST IS RUN OUT OF A DIFFERENT ADDRESS SPACE.
                                ;*   R0 CONTAINS THE ADDRESS UNDER TEST.

                                ;;*****************************************************************
030674  012737  000214  177746  TST41:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
030702  000001                          SCOPE
030704  012737  031132  001234          MOV     #TST42,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
030712  012737  000200  177746          MOV     #200,@#CCR      ;CACHE ON
030720  012702  060000                  MOV     #BUFL,R2        ;INIT REG FOR TEST ADDRESS
030724  012701  002000                  MOV     #2000,R1        ;INIT LOOP COUNT

                                        ;DOES THE TEST ADDR OVERLAP THE SAME ADDR SPACE IN CACHE AS THE PROGRAM INSTRUCT
```

```

030730  010237  001172          T25L04: MOV     R2,$TMP0        ;GET TEST ADDR.
030734  012737  031006  001174          MOV     #16,$TMP1       ;GET PROGRAM TEST INSTRUCTION ADDR.
030742  042737  174000  001172          BIC     #174000,$TMP0   ;CALC ADDRESSES CORRESP. CACHE ADDR
030750  042737  174000  001174          BIC     #174000,$TMP1   ;CALC ADDRESSES CORRESP. CACHE ADDR
030756  023737  001174  001172          CMP     $TMP1,$TMP0     ;DO THE CACHE ADDRESSES OVERLAP?
030764  101010                          BHI     1$              ;BRANCH IF NO
030766  062737  000012  001174          ADD     #12,$TMP1       ;CALC LAST PROG. TEST INSTRUCTION ADDR
030774  023737  001172  001174          CMP     $TMP0,$TMP1     ;DO THE CACHE ADDRESSES OVERLAP?
031002  101415                          BLOS    T25L01          ;BRANCH IF YES

031004  005012                          CLR     (R2)            ;PUT THE DATA IN CACHE
031006                          1$:
031006  012700  000200                  MOV     #200,R0         ;SET BIT IN R0 FOR CONSOLE CACHE SWEEP
031012  076600                          MED                     ;CONSOLE CACHE SWEEP
031014  000352                          .WORD   WINIT
031016  005712                          TST     (R2)            ;SEE IF LOC STILL IN CACHE
031020  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT?
031026  001016                          BNE     T25L02          ;BRANCH TO ERROR IF YES
031030  005722                  T25L03: TST     (R2)+           ;UPDATE ADDRESS
031032  077142                          SOB     R1,T25L04       ;BRANCH IF ALL CACHE NOT TESTED
031034  000436                          BR      TST42           ;;GO TO NEXT TEST

031036  005012                  T25L01: CLR     (R2)            ;PUT DATA IN CACHE
031040  012700  000200                  MOV     #200,R0         ;SET BIT IN R0 FOR CONSOLE CACHE SWEEP
031044  076600                          MED                     ;CONSOLE CACHE SWEEP
031046  000352                          .WORD   WINIT
031050  005712                          TST     (R2)            ;SEE IF LOC STILL IN CACHE
031052  033727  177752  000004          BIT     @#HMR,#HMR2     ;HIT?
031060  001001                          BNE     T25L02          ;BRANCH TO ERROR IF YES
031062  000762                          BR      T25L03          ;LOOK AT NEXT ADDRESS

031064  052737  000014  177746  T25L02: BIS     #14,@#CCR       ;CACHE OFF
031072  005037  001160                  CLR     $REG1           ;SAVE FAILING ADDRESS
031076  010237  001162                  MOV     R2,$REG2        ;SAVE FAILING ADDRESS
031102  012737  030730  001110          MOV     #T25L04,@#$LPERR  ;INIT RETURN FOR ERROR LOOP
031110  104113                          ERROR   113             ;ERROR: ADDR. NOT INVALIDATED BY CONSOLE SWEEP
031112  123727  001103  000003          CMPB    @#$ERFLG,#3     ;MORE THAN 3 ERRORS?
031120  101004                          BHI     TST42           ;;GO TO NEXT TEST IF YES
031122  012737  000200  177746          MOV     #200,@#CCR      ;CACHE ON
031130  000737                          BR      T25L03          ;CONTINUE TEST


                                ;;*****************************************************************
                                ;*TEST 42        TEST POWER UP INVALIDATES CACHE AND CLEARS CACHE CONTROL REG
                                ;*
                                ;*   THIS TEST IS ONLY RUN IF SW07=1.  THIS IS BECAUSE
                                ;*OPERATOR INTERVENTION IS NEEDED TO POWER DOWN AND THEN
                                ;*UP THE MACHINE WHEN THE MESSAGE IS TYPED ON THE TTY.
                                ;*IF RUNNING UNDER APT AND SW07=1,THE PROGRAM ASSUMES
                                ;*THAT A UNIBUS EXERCISOR (M7955) IS AVAILABLE
                                ;*TO POWER DOWN AND THEN UP THE MACHINE.
                                ;*AFTER THE MACHINE HAS DONE THIS, THE CACHE CONTROL REG
                                ;*IS EXAMINED TO HAVE BEEN PROPERLY INITIALIZED BY POWER
                                ;*UP.  AFTER THIS ALL CACHE IS REFERENCED.  THERE IS A
                                ;*VERY HIGH PROBABILITY THAT CACHE WILL HAVE PARITY ERRORS
                                ;*IF THE POWER UP FAILED TO SWEEP CACHE.  ANY CACHE PARITY
```

```
                                ;*ERROR THEREFORE IS REPORTED AS THE POWER UP FAILING TO
                                ;*INVALIDATE CACHE.IT SHOULD BE POINTED OUT THAT
                                ;*THE SWEEP MECHANISM IS CHECKED IN THE PREVIOUS TEST,THIS
                                ;*TEST VERIFIES THAT THE MECHANISM CAN BE INITIATED BY
                                ;*THE POWER UP SEQUENCE.
                                ;*
                                ;*NOTE:IF MACHINE HAS VOLATILE MEMORY, THE SWITCH
                                ;*     SETTINGS WILL HAVE TO BE RESTORED AFTER THIS TEST


                                ;;*****************************************************************
031132  012737  000214  177746  TST42:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
031140  000004                          SCOPE
031142  012737  031524  001234          MOV     #TST43,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
031150  032777  000200  147756          BIT     #SW07,@SWR      ;RUN THIS TEST?
031156  001563                          BEQ     TST43           ;;BRANCH TO NEXT TEST IF NO
031160  012737  031412  000114          MOV     #T20L01,@#PVEC  ;SET UP FOR PARITY ERRORS
031166  012737  031226  000024          MOV     #T20L02,@#PWRVEC ;SET UP FOR POWER DOWN
031174  012737  000314  177746          MOV     #314,@#CCR      ;SET ALL BITS IN CCR
031202  105737  001256                  TSTB    @#$ENV          ;RUNNING UNDER APT?
031206  001404                          BEQ     1$              ;BRANCH IF NO
031210  012777  000020  147776          MOV     #20,@CREG2      ;SET UP UBE TO POWER FAIL
031216  000777                          BR      .               ;WAIT FOR POWER FAIL

031220  104401  040625          1$:     TYPE    ,MSG2           ;POWER DOWN AND THEN UP
031224  000777                          BR      .               ;WAIT FOR POWER DOWN

031226  012737  031246  000024  T20L02: MOV     #T20L03,@#PWRVEC ;SET UP FOR POWER UP
031234  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
031236  017737  147672  001172          MOV     @SWR,$TMP0      ;SAVE (SWR)
031244  000777                          BR      .               ;WAIT FOR POWER UP

031246  012706  001100          T20L03: MOV     #STACK,SP       ;RESTORE STACK
031252  105737  001256                  TSTB    @#$ENV          ;RUNNING UNDER APT?
031256  001402                          BEQ     1$              ;BRANCH IF NO
031260  005077  147730                  CLR     @CREG2          ;STOP UBE POWER FAIL

031264  013700  001172          1$:     MOV     $TMP0,R0        ;GET (SWR)
031270  076600                          MED                     ;RESTORE SWR
031272  000226                          .WORD   WSW
031274  012701  177000                  MOV     # 177000,R1             ;INIT DELAY
031300  012700  177400          3$:     MOV     #177400,R0      ;INIT DELAY COUNTER FOR TTY
031304  063737  060000  060000  2$:     ADD     @#BUFL,@#BUFL   ;DELAY
031312  005200                          INC     R0
031314  001373                          BNE     2$              ;WAIT FOR TTY
031316  005201                          INC     R1
031320  001367                          BNE     3$              ;CONTINUE DELAY
031322  013737  177746  001160          MOV     @#CCR,$REG1     ;SEE IF CCR INITIALIZED
031330  001401                          BEQ     T20L04          ;BRANCH IF CCR CLEARED
031332  104101                          ERROR   101             ;ERROR: CACHE CONTROL REG NOT INIT BY POWER FAIL

031334                          T20L04:

031334  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
031336  076600                          MED                     ;GET CONTENTS OF LOG REG
031340  000022                          .WORD   RLOG
```

```
031342  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
031346  076600                          MED                     ;UNLOCK ERROR LOG
031350  000222                          .WORD   WLOG
031352  012600                          MOV     (SP)+,R0        ;RESTORE R0

031354  012737  000200  177746          MOV     #200,@#CCR      ;JAMUPP ON PARITY ERRORS
031362  012701  002000                  MOV     #2000,R1        ;INIT LOOP COUNT
031366  005000                          CLR     R0              ;INIT ADDRESS
031370  005720                  1$:     TST     (R0)+           ;REFERENCE ALL CACHE LOC
031372  077102                          SOB     R1,1$           ;LOOP TILL DONE
031374  012737  033142  000114  T20L06: MOV     #UPERR,@#PVEC   ;RESTORE PARITY ERROR HANDLER
031402  012737  040364  000024          MOV     #@PWRDN,@#PWRVEC ;RESTORE POWER FAIL HANDLER
031410  000445                          BR      TST43           ;;GO TO NEXT TEST

031412  052737  000014  177746  T20L01: BIS     #14,@#CCR       ;CACHE OFF TO STOP FURTHER PARITY ERRORS

031420  010046                          MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
031422  076600                          MED                     ;GET CONTENTS OF LOG REG
031424  000022                          .WORD   RLOG
031426  052700  100001                  BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
031432  076600                          MED                     ;UNLOCK ERROR LOG
031434  000222                          .WORD   WLOG
031436  012600                          MOV     (SP)+,R0        ;RESTORE R0

031440  011637  001164                  MOV     (SP),$REG3      ;GET PC+2 OF ERROR
031444  162737  000002  001164          SUB     #2,$REG3        ;SAVE PC OF ERROR
031452  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
031454  076600                          MED                     ;GET LOG INFORMATION
031456  000100                          .WORD   RJAM
031460  032700  000400                  BIT     #400,R0         ;ERROR IN BACKING STORE?
031464  001415                          BEQ     T20L05          ;BRANCH IF NO
031466  076600                          MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
031470  000101                          .WORD   RSER
031472  000300                          SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
031474  042700  177776                  BIC     #177776,R0      ;ONLY LOOK AT A17, A16
031500  010037  001160                  MOV     R0,$REG1        ;SAVE ADDRESS
031504  076600                          MED                     ;GET LOG INFORMATION
031506  000102                          .WORD   LOADD
031510  010037  001162                  MOV     R0,$REG2        ;SAVE INFORMATION
031514  104001                          ERROR   1               ;UNEXPECTED PARITY ERROR IN BACKING STORE
031516  000726                          BR      T20L06          ;GO TO NEXT TEST

031520  104102                  T20L05: ERROR   102             ;ERROR: POWER UP FAILED TO INVALIDATE CACHE
031522  000724                          BR      T20L06          ;GO TO NEXT TEST


                                ;;*************************************************************



                                        ;THE FOLLOWING TESTS ARE RUN ONLY IF SW08=1 AT THE
                                        ;BEGINNING OF THE PROGRAM, AND THE NPR DEVICE WAS SELECTED.
                                        ;THESE TESTS USE DATA SUPPLIED BY THE USER WHEN THE PROGRAM
                                        ;IS STARTED TO SETUP VARIOUS CONTROL REGISTERS TO RUN THE
                                        ;NPR DEVICE.  CREG1 ALWAYS CONTAINS THE DEVICES GO ADDRESS,
                                        ;EAD CONTAINS THE DEVICES ERROR REG ADDRESS, IVEC CONTAINS
                                        ;THE DEVICE'S INTERRUPT VECTOR (IF USED), SETUP CONTAINS THE
```

```
                                        ;ADDRESS OF THE DEVICE'S HANDLER AND THE REMAINING CREG2-5
                                        ;CONTAIN VARIOUS CONTROL INFORMATION NEEDED FOR THE PARTICULAR
                                        ;DEVICE USED.  THE SETUP HANDLERS ARE USED TO INITIALIZE THE
                                        ;DEVICE TO DO NPR DATOS TO THE STARTING ADDRESS FOLLOWING
                                        ;THE SUBROUTINE CALL.

                                ;;*************************************************************


                                ;;*************************************************************
                                ;*TEST 43       TEST NPR DATO INVALIDATES CACHE FOR PHYSICAL ADDRESS BITS A1-A10
                                ;*
                                ;*   THIS TEST IS ONLY RUN IF SW08=1.  THE PREVIOUSLY SEL-
                                ;*ECTED DEVICE IS SETUP TO DO NPR DATO'S TO ADDRESSES IN
                                ;*A 1K BUFFER AREA.  ON THE FIRST PASS A '1' IS FLOATED
                                ;*THROUGH THE ADDRESS LINES A1-A10.  AT EACH BIT POSITION,
                                ;*THE LOC IS PUT IN CACHE AND THEN A NPR DATO IS DONE TO
                                ;*THAT LOC.  A MINIMUM TIME IS THEN WAITED TO ALLOW THE
                                ;*SLOWEST DEVICE SELECTABLE TO FINISH ITS TRANSFERS.  THE
                                ;*LOC IS THEN CHECKED TO BE A MISS.
                                ;*   FOR THE SECOND PASS, A '0' IS FLOATED THROUGH ADDRESS
                                ;*BITS A1-A10 AND THE SAME PROCEDURE IS REPEATED.  BEFORE
                                ;*THE DEVICE'S GO BIT IS SET, THE TRANSFER ADDRESS IS CHECKED
                                ;*TO SEE IF IT OVERLAPS THE INSTRUCTION ADDRESS IN CACHE.
                                ;*IF IT DOES, THE INSTRUCTIONS ARE EXECUTED OUT OF A NON
                                ;*OVERLAPPING ADDRESS SPACE.
                                ;*    R0  CONTAINS THE DEVICES GO ADDRESS
                                ;*    R1  CONTAINS THE PASS INDICATOR
                                ;*    R2  IS THE DELAY COUNTER
                                ;*    R3  CONTAINS THE TRANSFER ADDRESS
                                ;*    R4  USED TO CALCULATE NEXT TRANSFER ADDRESS

                                ;;*************************************************************
031524  012737  000214  177746  TST43:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
031532  000004                          SCOPE
031534  012737  032240  001234          MOV     #TST44,$KTST    ;SAVE POINTER TO NEXT TEST IF UNEXPECTED PARITY ERROR
031542  032777  000400  147364          BIT     #SW08,@SWR      ;NPR DEVICE AVAILABLE?
031550  001002                          BNE     1$              ;BRANCH IF YES
031552  000137  032240                  JMP     @#TST44         ;NO GO TO NEXT TEST
031556  012737  000200  177746  1$:     MOV     #200,@#CCR      ;CACHE ON
031564  004737  034000                  JSR     PC,VEC          ;SEE IF UBE NEW USED AND SETUP INTERRUPT VECTOR
031570  013700  001212                  MOV     CREG1,R0        ;GET DEVICE'S GO ADDRESS
031574  005001                          CLR     R1              ;CLEAR FLAG FOR PASS 1 (FLOAT 1 PATTERN)
031576  012704  000002                  MOV     #2,R4           ;INIT REG FOR ADDR. CALC.
031602  012737  060002  031620          MOV     #BUFL+2,@#ADD1L ;INIT ADDRESS LOWER FOR TEST
031610  005037  031622                  CLR     @#ADD1H         ;INIT ADDRESS HIGHER FOR TEST
031614  004777  147412          T21L09: JSR     PC,@SETUP       ;SETUP DEVICE TO DO NPR DATO TO FOLLOWING ADDRESS
031620  000000                  ADD1L:  .WORD   0               ;TEST ADDRESS LOWER 16 BITS
031622  000000                  ADD1H:  .WORD   0               ;TEST ADDRESS UPPER 2 BITS
031624  005002                          CLR     R2              ;INIT R2 FOR TIME DELAY COUNT

                                        ;FIND OUT IF THE TEST INSTRUCTION ADDRESS IN CACHE
                                        ;OVERLAP THE XFER ADDRESS IN CACHE.  IF THEY DO, USE THE
                                        ;TEST INSTRUCTIONS AT NON OVERLAPPING ADDRESS.  THIS IS TO
                                        ;ENSURE THAT A MISS IS DUE TO A INVALIDATE RATHER THAN
                                        ;THE TEST INSTRUCTION SWAPPING OUT OF CACHE THE XFER LOCATION.
```

```

031626  013737  031620  001172          MOV     ADD1L,$TMP0     ;GET XFER ADDRESS
031634  042737  174000  001172          BIC     #174000,$TMP0   ;CALC ITS CACHE ADDRESS
031642  012737  031712  001174          MOV     #T21L01,$TMP1   ;GET TEST INST ADDRESS
031650  042737  174000  001174          BIC     #174000,$TMP1   ;CALC ITS CACHE ADDRESS
031656  023737  001172  001174          CMP     $TMP0,$TMP1     ;DOES XFER ADDRESS OVERLAP TEST INST?
031664  002407                          BLT     T21L02          ;BRANCH IF NO
031666  062737  000022  001174          ADD     #22,$TMP1       ;GET ADDRESS OF LAST OVERLAPPING TEST INST.
031674  023737  001172  001174          CMP     $TMP0,$TMP1     ;DOES XFER ADDRESS STILL OVERLAP TEST INST?
031702  003503                          BLE     T21L03          ;BRANCH IF YES TO TEST INST. AT DIFFERENT CACHE ADDRESS
031704  013703  031620          T21L02: MOV     ADD1L,R3        ;GET XFER ADDRESS
031710  021313                          CMP     (R3),(R3)       ;MAKE XFER ADDRESS A HIT
031712  033727  177752  000004  T21L01: BIT     @#HMR,#HMR2     ;MAKE SURE ITS IN CACHE
031720  001514                          BEQ     T21L04          ;BRANCH IF NO TO ERROR
031722  005210                          INC     (R0)            ;SET DEVICES GO BIT TO START DATA XFERS
031724  005202                  1$:     INC     R2              ;DELAY TILL THE SLOWEST DEVICE
031726  001376                          BNE     1$              ;HAS FINISHED ITS XFERS
031730  005713                          TST     (R3)            ;SEE IF NPR DATO HAS INVALIDATED THE XFER ADDRESS IN CAC
031732  033727  177752  000004          BIT     @#HMR,#HMR2     ;LOC NOW A MISS? (CACHE INVALIDATED?)
031740  001117                          BNE     T21L05          ;GO REPORT ERROR IF LOC A HIT
031742  005777  147262          T21L11: TST     @EAD            ;SEE IF DEVICE HAD AN ERROR
031746  100514                          BMI     T21L05          ;REPORT DEVICE ERROR IF YES
031750  005701                          TST     R1              ;PASS 1?
031752  001024                          BNE     T21L07          ;BRANCH IF NO
031754  032704  002000                  BIT     #2000,R4        ;LAST FLOAT 1 PATTERN USED?
031760  001007                          BNE     T21L08          ;BRANCH IF YES
031762  006304                          ASL     R4              ;GENERATE NEXT FLOAT 1 PATTERN
031764  010437  031620                  MOV     R4,ADD1L        ;SAVE ITS LOWER BITS
031770  052737  060000  031620          BIS     #BUFL,ADD1L     ;SET ITS HIGH BITS SO ITS IN TEST BUFFER
031776  000706                          BR      T21L09          ;GO TEST IT

032000  052701  000001          T21L08: BIS     #1,R1           ;SET FLAG FOR PASS 2 TO INDICATE FLOAT 0 PATTERN
032004  012704  001776                  MOV     #1776,R4        ;INIT REG FOR TEST ADDR. CALC.
032010  010437  031620                  MOV     R4,ADD1L        ;SAVE LOWER TEST ADDR.
032014  052737  060000  031620          BIS     #BUFL,ADD1L     ;MAKE SURE ADDR. IN TEST AREA
032022  000674                          BR      T21L09          ;GO TEST IT

032024  022704  003776          T21L07: CMP     #3776,R4        ;AT LAST FLOAT 0 PATTERN?
032030  001413                          BEQ     T21L10          ;BRANCH IF YES TO END OF TEST
032032  006204                          ASR     R4              ;GENERATE NEW TEST ADDR.
032034  052704  002000                  BIS     #2000,R4        ;MAKE IT A FLOAT 0 PATTERN
032040  042704  000001                  BIC     #1,R4           ;MAKE IT A WORD ADDR.
032044  010437  031620                  MOV     R4,ADD1L        ;SAVE LOWER TEST ADDR.
032050  052737  060000  031620          BIS     #BUFL,ADD1L     ;MAKE SURE ADDRESS IS IN TEST BUFFER
032056  000656                          BR      T21L09          ;GO TEST IT

032060  022737  034046  001232  T21L10: CMP     #HUBEN,SETUP    ;NEW UNIBUS EXERCISOR USED?
032066  001064                          BNE     TST44           ;;BRANCH TO NEXT TEST IF NO
032070  013737  001226  001172          MOV     IVEC,$TMP0      ;GET UBE INTERRUPT VECTOR
032076  062737  000002  001172          ADD     #2,$TMP0        ;AND RESTORE
032104  005077  147062                  CLR     @$TMP0          ;THE TRAP CATCHER
032110  000453                          BR      TST44           ;;GO TO NEXT TEST

                                        ;TEST INST TO TEST XFER ADDRESSES WHICH OVERLAP TEST CODE

032112  013703  031620          T21L03: MOV     ADD1L,R3        ;GET XFER ADDRESS
```

```
032116  021313                          CMP     (R3),(R3)       ;MAKE ADDRESS A HIT
032120  033727  177752  000004          BIT     @#HMR,#HMR2     ;MAKE SURE ITS IN CACHE
032126  001411                          BEQ     T21L04          ;BRANCH IF NO TO ERROR
032130  005210                          INC     (R0)            ;SET DEVICES GO BIT TO START DATA XFER
032132  005202                  1$:     INC     R2              ;DELAY TILL THE SLOWEST DEVICE
032134  001376                          BNE     1$              ;HAS FUNISHED ITS XFERS.
032136  005713                          TST     (R3)            ;SEE IF NPR DATO HAS INVALIDATED XFER ADDR. IN CACHE
032140  033727  177752  000004          BIT     @#HMR,#HMR2     ;IS LOC NOW A MISS? (CACHE INVALIDATED?)
032146  001014                          BNE     T21L05          ;GO REPORT ERROR IF LOC A HIT
032150  000674                          BR      T21L11          ;CHECK FOR DEVICE ERROR

032152  012737  031614  001110  T21L04: MOV     #T21L09,@#LPERR ;SET UP RETURN FOR ERROR LOOP
032160  013737  031622  001160          MOV     ADD1H,$REG1     ;SAVE 'BAD' ADDRESS
032166  013737  031620  001162          MOV     ADD1L,$REG2     ;SAVE 'BAD' ADDRESS
032174  104043                          ERROR   43              ;ERROR: ADDRESS COULD NOT BE MADE A HIT
032176  000730                          BR      T21L10          ;GO TO END OF TEST

032200  012737  031614  001110  T21L05: MOV     #T21L09,@#LPERR ;SET UP RETURN FOR ERROR LOOP
032206  013737  031622  001160          MOV     ADD1H,$REG1     ;SAVE BAD ADDRESS
032214  013737  031620  001162          MOV     ADD1L,$REG2     ;SAVE BAD ADDRESS
032222  005777  147002                  TST     @EAD            ;DID DEVICE HAVE ERROR?
032226  100002                          BPL     1$              ;BRANCH IF NO
032230  104103                          ERROR   103             ;ERROR: DEVICE ERROR BIT SET WHEN DOING DATO TO ADDRESS
032232  000712                          BR      T21L10          ;GO TO END OF TEST
032234  104104                  1$:     ERROR   104             ;ERROR: CACHE LOC NOT INVALIEDATED BY NPR DATO TO ADDR.
032236  000710                          BR      T21L10          ;GO TO END OF TEST.

                                ;;*************************************************************
                                ;*TEST 44       TEST NPR DATO INVALIDATES CACHE FOR PHYSICAL ADDRESS BITS A17-A11
                                ;*
                                ;*   THIS TEST IS RUN ONLY IF SW00=1 AND THE INHIBIT TESTS
                                ;*USING KT (SW12)=0.  THE PREVIOUSLY SELECTED DEVICE IS
                                ;*SETUP TO DO NPR DATOS TO LOCATIONS LYING OUTISDE THE
                                ;*PROGRAM AND MONITOR ADDRESS SPACE.  (THE MONITOR, IF IT
                                ;*EXISTS, LIES IN THE LAST 1.5K OF MEMORY).
                                ;*   MEMORY IS FIRST SIZED TO DETERMINE THE MAXIMUM TESTABLE
                                ;*ADDRESS.A VIRTUAL  ADDRESS IS GENERATED AND STORED IN KIPAR4.
                                ;*THEN ITS PHYSICAL ADDRESS IS CALCULATED FOR THE DEVICES
                                ;*NPR TRANSFER.  THE ADDRESS IS MADE A HIT IN CACHE AND THEN
                                ;*AN NPR DATO IS DONE TO IT.  A MINIMUN TIME IS THEN WAITED
                                ;*TO ALLOW THE SLOWEST SELECTABLE DEVICE TO FINISH ITS
                                ;*TRANSFERS.  THE LOCATION IS THEN CHECKED TO BE A MISS
                                ;*(INVALIDATED).  A NEW TAG VALUE IS THEN GENERATED AND
                                ;*THE PROCEDURE REPEATS TO THE MAXIMUM ALLOWABLE ADDRESS.
                                ;*   BEFORE THE DEVICE'S GO BIT IS SET, THE ADDRESS IS
                                ;*CHECKED TO SEE IF IT OVERLAPS THE INSTRUCTION ADDRESSES
                                ;*IN CACHE.  IF IT DOES, THE INSTRUCTIONS ARE EXECUTED
                                ;*OUT OF  NON OVERLAPPING ADDRESSES.
                                ;*
                                ;*   R0  CONTAINS THE DEVICE'S GO ADDRESS
                                ;*   R2  IS THE DELAY COUNTER
                                ;*   R3  IS USED TO GENERATE THE TEST ADDRESS

                                ;;*************************************************************
032240  012737  000214  177746  TST44:  MOV     #214,@#CCR      ;CACHE OFF FOR SCOPE
032246  000004                          SCOPE
```

```
 032250  012737  033020  001234            MOV     #$EOP,$KTST
 032256  032777  000400  146650            BIT     #SW08,@SWR        ;NPR DEVICE AVAILABLE?
 032264  001002                            BNE     1$                ;BRANCH IF YES
 032266  000137  033020                    JMP     @#$EOP
 032272  032777  010000  146634  1$:       BIT     #SW12,@SWR        ;INHIBIT TESTS USING KT?
 032300  001402                            BEQ     2$                ;BRANCH IF NO
 032302  000137  033020                    JMP     @#$EOP
 032306  012737  000200  177746  2$:       MOV     #200,@#CCR        ;CACHE ON
 032314  052737  000200  036034            BIS     #200,@#$KT11      ;USE KT FOR $SIZE
 032322  004737  035750                    JSR     PC,$SIZE          ;SIZE MEM
 032326  162737  000040  036322            SUB     #40,$LSTBK        ;REDUCE TESTABLE MEM BY 1K SO DON'T KILL MONITOR IF EXIS
 032334  013700  001212                    MOV     CREG1,R0          ;GET THE GO ADDRESS OF THE DEVICE
 032340  005237  177572                    INC     @#MMR0            ;TURN KT ON

                                           ;CALC TEST ADDR

 032344  012703  000020                    MOV     #20,R3            ;INIT ADDR REG
 032350  006303                  T22L00:   ASL     R3                ;CALC NEXT ADDR
 032352  010337  172350                    MOV     R3,@#KIPAR4       ;SETUP PAR WITH TEST ADDR
 032356  023727  172350  001000            CMP     @#KIPAR4,#1000    ;PAST INSTRUCTION SPACE?
 032364  002003                            BGE     1$                ;BRANCH IF YES
 032366  052737  000600  172350            BIS     #600,@#KIPAR4     ;MAKE SURE TEST ADDR, LIES OUTSIDE OF TEST CODE
 032374  023737  172350  036322  1$:       CMP     @#KIPAR4,$LSTBK ;IS TEST ADDRESS IN MONITOR ADDRESS SPACE?
 032402  003164                            BGT     T22L02            ;BRANCH IF YES TO END OF TEST
 032404  004737  034000                    JSR     PC,VEC            ;SET UP UBE NEW INT. VECT IF IT IS USED

                                           ;CALC THE PHYSICAL ADDRESS FROM THE VIRTUAL TEST ADDRESS

 032410  012737  101776  001172            MOV     #101776,$TMP0     ;GET VIRTUAL ADDRESS
 032416  004737  033434                    JSR     PC,VIP            ;CALC ITS PHYSICAL ADDRESS
 032422  013737  001160  032444            MOV     $REG1,ADD2H       ;SAVE PHYSICAL TEST ADDRESS
 032430  013737  001162  032442            MOV     $REG2,ADD2L       ;SAVE PHYSICAL TEST ADDRESS
 032436  004777  146570          T22L11:   JSR     PC,@SETUP         ;SETUP NPR DEVICE TO DO DATO TO FOLLOWING ADDRESS
 032442  000000                  ADD2L:    .WORD   0                 ;TEST ADDRESS LOWER 16 BITS
 032444  000000                  ADD2H:    .WORD   0                 ;TEST ADDRESS UPPER 2 BITS
 032446  005002                            CLR     R2                ;INIT REG FOR TIME DELAY

                                           ;FIND OUT IF THE TEST INSTRUCTION ADDRESS IN CACHE
                                           ;OVERLAP THE XFER ADDRESS IN CACHE.  IF THEY DO, USE
                                           ;TEST INSTRUCTIONS AT NON OVERLAPPING ADDRESS.  THIS IS TO
                                           ;ENSURE THAT A MISS IS DUE TO A INVALIDATE RATHER THAN
                                           ;THE TEST INSTRUCTION SWAPPING OUT OF CACHE THE XFER LOCATION.

 032450  013737  032442  001172            MOV     ADD2L,$TMP0       ;GET XFER ADDRESS
 032456  042737  174000  001172            BIC     #174000,$TMP0     ;CALC ITS CACHE ADDRESS
 032464  012737  032534  001174            MOV     #T22L01,$TMP1     ;GET TEST INST ADDRESS
 032472  042737  174000  001174            BIC     #174000,$TMP1     ;CALC ITS CACHE ADDRESS
 032500  023737  001172  001174            CMP     $TMP0,$TMP1       ;DOES XFER ADDRESS OVERLAP TEST INST?
 032506  002407                            BLT     T22L03            ;BRANCH IF NO
 032510  062737  000024  001174            ADD     #24,$TMP1         ;CALC ADDR OF LAST OVERLAPPING TEST INST.
 032516  023737  001172  001174            CMP     $TMP0,$TMP1       ;DOES XFER ADDR STILL OVERLAP TEST INST?
 032524  003440                            BLE     T22L04            ;BRANCH IF YES

 032526  023737  101776  101776  T22L03:   CMP     @#101776,@#101776  ;MAKE ADDR A HIT
 032534  033727  177752  000004  T22L01:   BIT     @#HMR,#HMR2       ;MAKE SURE ITS IN CACHE
 032542  001452                            BEQ     T22L05            ;BRANCH IF NO TO ERROR
```

```
 032544  005210                            INC     (R0)              ;SET DEVICE'S GO BIT TO DO DATA XFERS
 032546  005202                  1$:       INC     R2                ;DELAY TILL THE SLOWEST DEVICE
 032550  001376                            BNE     1$                ;HAS FINISHED ITS XFERS
 032552  005737  101776                    TST     @#101776          ;SEE IF NPR DATO HAS INVALIDATED THE XFER ADDR IN CACHE
 032556  033727  177752  000004            BIT     @#HMR,#HMR2       ;LOC NOW A MISS? (CACHE INVALIDATED?)
 032564  001054                            BNE     T22L06            ;GO REPORT ERROR IF LOC A HIT
 032566  005777  146436          T22L10:   TST     @EAD              ;SEE IF DEVICE HAS AN ERROR
 032572  100451                            BMI     T22L06            ;REPORT DEVICE ERROR IF YES
 032574  023727  172350  004000            CMP     @#KIPAR4,#4000    ;TESTED LAST ADDRESS?
 032602  001464                            BEQ     T22L02            ;BRANCH TO END OF TEST IF YES
 032604  022737  034174  001232            CMP     #NUBEO,SETUP      ;WAS THE OLD UBE USED?
 032612  001256                            BNE     T22L00            ;NO, GO CALC NEXT TEST ADDR
 032614  023727  172350  001000            CMP     @#KIPAR4,#1000    ;AT LAST TESTABLE ADDRESS FOR OLD UBE?
 032622  002652                            BLT     T22L00            ;BRANCH IF NO
 032624  000453                            BR      T22L02            ;GO TO END OF TEST

 032626  023737  101776  101776  T22L04:   CMP     @#101776,@#101776  ;MAKE XFER ADDRESS A HIT
 032634  033727  177752  000004            BIT     @#HMR,#HMR2       ;MAKE SURE ITS IN CACHE
 032642  001412                            BEQ     T22L05            ;BRANCH TO ERROR IF NO
 032644  005210                            INC     (R0)              ;SET DEVICES TO BIT TO DO XFERS.
 032646  005202                  1$:       INC     R2                ;DELAY TILL THE SLOWEST DEVICE
 032650  001376                            BNE     1$                ;HAS FINISHED
 032652  005737  101776                    TST     @#101776          ;SEE IF NPR DATO HAS INVALID THE XFER ADDR. IN CACHE
 032656  033727  177752  000004            BIT     @#HMR,#HMR2       ;LOC NOW A MISS? (CACHE INVALIDATED?)
 032664  001014                            BNE     T22L06            ;GO REPORT ERROR IF LOC A HIT
 032666  000737                            BR      T22L10            ;GO SEE IF DEVICE HAD AN ERROR

 032670  012737  032436  001110  T22L05:   MOV     #T22L11,@#$LPERR          ;INIT RETURN FOR ERROR LOOP
 032676  013737  032444  001160            MOV     ADD2H,$REG1       ;SAVE BAD ADDRESS
 032704  013737  032442  001162            MOV     ADD2L,$REG2       ;SAVE BAD ADDRESS
 032712  104043                            ERROR   43                ;ERROR: ADDRESS COULD NOT BE MADE A HIT
 032714  000417                            BR      T22L02            ;GO TO END OF TEST

 032716  012737  032436  001110  T22L06:   MOV     #T22L11,@#$LPERR  ;SETUP RETURN FOR ERROR LOOPS
 032724  013737  032444  001160            MOV     ADD2H,$REG1       ;SAVE BAD ADDRESS
 032732  013737  032442  001162            MOV     ADD2L,$REG2       ;SAVE BAD ADDRESS
 032740  005777  146264                    TST     @EAD              ;DID DEVICE HAVE AN ERROR?
 032744  100002                            BPL     1$                ;BRANCH IF NO
 032746  104103                            ERROR   103               ;ERROR: DEVICE ERROR BIT SET WHEN DOING DATO TO ADDR.
 032750  000401                            BR      T22L02            ;GO TO END OF TEST

 032752  104104                  1$:       ERROR   104               ;ERROR: CACHE LOC NOT INVALID BY NPR DATO TO ADDR.
 032754  042737  000001  177572  T22L02:   BIC     #1,@#MMR0         ;KT OFF
 032762  023727  001232  014046            CMP     SETUP,#NUBEN      ;WAS THE NEW UBE USED?
 032770  001013                            BNE     $EOP              ;IF NO, GO TO END-OF-PASS
 032772  013737  001226  001172            MOV     IVEC,$TMP0        ;GET UBE INTERRUPT VECTOR
 033000  062737  000002  001172            ADD     #2,$TMP0          ;AND RESTORE
 033006  013777  001172  146212            MOV     $TMP0,@IVEC
 033014  005077  146152                    CLR     @$TMP0            ;THE TRAP CATCHER



                                 .SBTTL  END OF PASS ROUTINE

                                 ;;*****************************************************************
                                 ;*INCREMENT THE PASS NUMBER ($PASS)
```

```
                                             ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                             ;*IF THERES A MONITOR GO TO IT
                                             ;*IF THERE ISN'T JUMP TO START1

 033020                                      $EOP:
 033020  000004                                      SCOPE
 033022  005037  001102                              CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 033026  005037  035406                              CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 033032  005237  001244                              INC     $PASS           ;;INCREMENT THE PASS NUMBER
 033036  042737  100000  001244                      BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 033044  005327                                      DEC     (PC)+           ;;LOOP?
 033046  000001                              $EOPCT: .WORD   1
 033050  003022                                      BGT     $DOAGN          ;;YES
 033052  012737                                      MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 033054  000001                              $ENDCT: .WORD   1
 033056  033046                                      $EOPCT
 033060  104401  033125                              TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 033064  013746  001244                              MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 033070  104405                                      TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 033072  104401  033122                              TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 033076  013700  000042                      $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 033102  001405                                      BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 033104  000005                                      RESET                   ;;CLEAR THE WORLD
 033106  004710                              $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 033110  000240                                      NOP                     ;;SAVE ROOM
 033112  000240                                      NOP                     ;;FOR
 033114  000240                                      NOP                     ;;ACT11
 033116                                      $DOAGN:
 033116  000137                                      JMP     @(PC)+          ;;RETURN
 033120  003056                              $RTNAD: .WORD   START1
 033122     377     377     000              $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 033125     015  042412  042116              $ENDMG: .ASCIZ  <15><12>/END PASS #/
 033132  050040  051501  020123
 033142  000043

                                             ;//////////////////////////////////////////////////////////////////////
                                             ;SUBROUTINE TO REPORT AN UNEXPECTED PARITY ERRORS
                                             ;//////////////////////////////////////////////////////////////////////

 033142  012737  000214  177746              UPERR:  MOV     #214,@#CCR      ;TURN OFF CACHE TO PREVENT OTHER ERRORS
 033150  011637  001166                              MOV     (SP),$REG4      ;SAVE PC+2 WHERE PARITY ERROR OCCURRED
 033154  162737  000002  001166                      SUB     #2,$REG4        ;CALC. PC WHERE PARITY ERROR OCCURRED
 033162  022626                                      CMP     (SP)+,(SP)+     ;RESTORE STACK
 033164  076600                                      MED                     ;GET LOG INFOR FOR PHY. ADDR. A17,A16
 033166  000101                                      .WORD   RBER
 033170  000300                                      SWAB    R0              ;PUT PHY. ADDR A17, A16 IN LOW BYTE
 033172  042700  177776                              BIC     #177776,R0      ;ONLY LOOK AT A17, A16
 033176  010037  001160                              MOV     R0,$REG1        ;SAVE ADDRESS
 033202  076600                                      MED                     ;GET LOG INFORMATION
 033204  000102                                      .WORD   LOADD
 033206  010037  001162                              MOV     R0,$REG2        ;SAVE INFORMATION
 033212  076600                                      MED                     ;GET LOG INFORMATION
 033214  000100                                      .WORD   RJAM
 033216  032700  000400                              BIT     #400,R0         ;WAS ERROR IN BACKING STORE?
 033222  001016                                      BNE     UP1             ;BRANCH IF YES
 033224  032737  000040  177744                      BIT     #40,@#EREG      ;WAS ERROR IN CACHE TAG FIELD?
```

```
 033232  001017                                      BNE     UP2             ;BRANCH IF YES
 033234  032737  000100  177744                      BIT     #100,@#EREG     ;WAS ERROR IN CACHE LOW BYTE?
 033242  001024                                      BNE     UP3             ;BRANCH IF YES
 033244  076600                                      MED                     ;GET LOG INFORMATION
 033246  000106                                      .WORD   CDH
 033250  010037  001164                              MOV     R0,$REG3        ;SAVE INFORMATION
 033254  104004                                      ERROR   4               ;ERROR: UNEXPECTED PARITY ERROR IN CACHE DATA HIGH
 033256  000423                                      BR      UPR             ;RETURN

 033260  013737  001166  001164              UP1:    MOV     $REG4,$REG3     ;SAVE PC OF TRAP
 033266  104001                                      ERROR   1               ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE
 033270  000416                                      BR      UPR             ;RETURN

 033272                                      UP2:
 033272  076600                                      MED                     ;GET TAG LOG INFO.
 033274  000107                                      .WORD   RTAG
 033276  000300                                      SWAB    R0              ;PUT TAG IN LOW BYTE
 033300  042700  177400                              BIC     #177400,R0      ;LOOK AT TAG ONLY
 033304  010037  001164                              MOV     R0,$REG3        ;SAVE CACHE TAG DATA
 033310  104002                                      ERROR   2               ;ERROR: UNEXPECTED PARITY ERROR IN CACHE TAG
 033312  000405                                      BR      UPR             ;RETURN

 033314                                      UP3:
 033314  076600                                      MED                     ;GET LOG INFORMATION
 033316  000106                                      .WORD   CDL
 033320  010037  001164                              MOV     R0,$REG3        ;SAVE INFORMATION
 033324  104003                                      ERROR   3               ;ERROR: UNEXPECTED PARITY ERROR IN CACHE DATA LOW

 033326                                      UPR:

 033326  010046                                      MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
 033330  076600                                      MED                     ;GET CONTENTS OF LOG REG
 033332  000022                                      .WORD   RLOG
 033334  052700  100001                              BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
 033340  076600                                      MED                     ;UNLOCK ERROR LOG
 033342  000222                                      .WORD   WLOG
 033344  012600                                      MOV     (SP)+,R0        ;RESTORE R0

 033346  000177  145662                              JMP     @SKTST          ;START SUBTEST FOLLOWING ONE WHERE ERROR OCCURRED
                                             ;//////////////////////////////////////////////////////////////////////
                                             ;ROUTINE TO REPORT UNEXPECTED TRAPS TO VECTOR 4
                                             ;//////////////////////////////////////////////////////////////////////
 033352  012737  000214  177746              UT4:    MOV     #214,@#CCR      ;TURN OFF CACHE
 033360  011637  001162                              MOV     (SP),$REG2      ;SAVE FAILING PC
 033364  013737  177766  001160                      MOV     @#CER,$REG1     ;GET CPU ERROR REG (CER)
 033372  032777  001000  145534                      BIT     #SW09,@SWR      ;LOOP ON ERROR?
 033400  001401                                      BEQ     1$              ;BRANCH IF NO
 033402  022626                                      CMP     (SP)+,(SP)+     ;RESTORE STACK
 033404  104016                              1$:     ERROR   16              ;ERROR: UNEXPECTED TRAP TO VECTOR 4

 033406  010046                                      MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
 033410  076600                                      MED                     ;GET CONTENTS OF LOG REG
 033412  000022                                      .WORD   RLOG
 033414  052700  100001                              BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
 033420  076600                                      MED                     ;UNLOCK ERROR LOG
 033422  000222                                      .WORD   WLOG
```

```
033424  012600                          MOV     (SP)+,R0        ;RESTORE R0

033426  000000                          HALT                    ;
033430  000137  000200                  JMP     @#200           ;RESTART TEST IF CONTINUE
                                ;//////////////////////////////////////////////////////////////////////
                                ;SUBROUTINE TO CONVERT VIRTUAL ADDRESS IN $TMP0 TO A PHYSICAL ADDRESS IN $REG2, $REG1
                                ;//////////////////////////////////////////////////////////////////////

033434  010146                  VIP:    MOV     R1,-(SP)        ;SAVE R1 ON STACK
033436  010246                          MOV     R2,-(SP)        ;SAVE R2 ON STACK
033440  013701  001172                  MOV     $TMP0,R1        ;GET VIRTUAL ADDRESS
033444  005002                          CLR     R2              ;INT SHIFT COUNTER
033446  006201                  1$:     ASR     R1              ;SHIFT BLOCK # TO LSB 0-6
033450  005202                          INC     R2              ;COUNT SHIFTS
033452  020227  000006                  CMP     R2,#6           ;ALL DONE?
033456  001373                          BNE     1$              ;BRANCH IF NO
033460  010137  001162                  MOV     R1,$REG2        ;SAVE BLOCK #
033464  042737  177600  001162          BIC     #177600,$REG2   ;MASK BLOCK #
033472  006201                  2$:     ASR     R1              ;SHIFT ACTIVE PAGE FIELD TO LSB 1-3
033474  005202                          INC     R2              ;COUNT SHIFTS
033476  020227  000014                  CMP     R2,#14          ;ALL DONE?
033502  001373                          BNE     2$              ;BRANCH IF NO
033504  042701  177761                  BIC     #177761,R1      ;CALC. APFX2
033510  062701  172340                  ADD     #KIPAR0,R1      ;CALC ADDRESS OF PAR REFERENCING
033514  011101                          MOV     (R1),R1         ;GET (PAR)
033516  060137  001162                  ADD     R1,$REG2        ;CALC. PHYSICAL BLOCK #
033522  013737  001162  001160          MOV     $REG2,$REG1     ;SAVE PHYSICAL ADDRESS BITS 17,16
033530  005002                          CLR     R2              ;INI. SHIFT COUNT
033532  006237  001160          3$:     ASR     $REG1           ;SHIFT ADDRESS BITS 17,16 TO LSB 1,0
033536  005202                          INC     R2              ;COUNT SHIFTS
033540  020227  000012                  CMP     R2,#12          ;DONE?
033544  001372                          BNE     3$              ;BRANCH IF NO
033546  005002                          CLR     R2              ;INIT. SHIFT COUNT
033550  006337  001162          4$:     ASL     $REG2           ;SHIFT MSB OF ADDRESS TO BIT 16
033554  005202                          INC     R2              ;COUNT SHIFTS
033556  020227  000006                  CMP     R2,#6           ;ALL DONE?
033562  001372                          BNE     4$              ;BRANCH IF NO
033564  013701  001172                  MOV     $TMP0,R1        ;GET VIRTUAL ADDRESS
033570  042701  177700                  BIC     #177700,R1      ;MASK OFF BLOCK COUNT
033574  060137  001162                  ADD     R1,$REG2        ;HAVE $REG2 CONTAIN PHYSICAL ADDRESS 0-15
033600  012602                          MOV     (SP)+,R2        ;RESTORE R2
033602  012601                          MOV     (SP)+,R1        ;RESTORE R1
033604  000207                          RTS     PC              ;RETURN

                                ;//////////////////////////////////////////////////////////////////////
                                        ;SUBROUTINE TO CALC. TAG FIELD FROM A PAR IN LOC $TMP0
                                ;//////////////////////////////////////////////////////////////////////

033606  010146                  TAG:    MOV     R1,-(SP)        ;SAVE R1 ON STACK
033610  012701  000005                  MOV     #5,R1           ;INIT R1 TO COUNT 5 SHIFTS
033614  006237  001172          1$:     ASR     $TMP0           ;CALC TAG CONTENTS
033620  077103                          SOB     R1,1$           ;ALL DONE?
033622  052737  000200  001172          BIS     #200,$TMP0      ;SET VALID BIT
033630  012601                          MOV     (SP)+,R1        ;RESTORE R1
033632  000207                          RTS     PC              ;RETURN

```

```
                                ;//////////////////////////////////////////////////////////////////////
                                        ;SUBROUTINE TO FIND PAR FROM A VIRTUAL ADDRESS IN R0 OR R1  AND
                                        ;PUT ITS CONTENTS IN $TMP0
                                ;//////////////////////////////////////////////////////////////////////
033634  010037  001172          PAR:    MOV     R0,$TMP0        ;GET VIRTUAL ADDRESS
033640  005776  000000                  TST     @(SP)           ;WAS R0 USED?
033644  001402                          BEQ     1$              ;BRANCH IF YES
033646  010137  001172                  MOV     R1,$TMP0        ;GET VIRTUAL ADDRESS
033652  062716  000002          1$:     ADD     #2,(SP)         ;ADJUST PC
033656  012705  000014                  MOV     #14,R5          ;INIT COUNT
033662  006237  001172          2$:     ASR     $TMP0           ;SHIFT ADDRESS TO GET ACTIVE PAGE FIELD
033666  077503                          SOB     R5,2$           ;APF IN LSB 1-3? BRANCH IF NO
033670  042737  177761  001172          BIC     #177761,$TMP0   ;MASK APF X 2
033676  062737  172340  001172          ADD     #KIPAR0,$TMP0   ;PUT PAR ADDRESS IN $TMP0
033704  017737  145262  001172          MOV     @$TMP0,$TMP0    ;GET CONTENTS OF PAR
033712  000207                          RTS     PC              ;RETURN
                                ;//////////////////////////////////////////////////////////////////////
                                        ;SUBROTINE TO GENERATE A TEST BUFFER ADDRESS 512(10) LOCATIONS FROM GIVEN
                                        ;ADDRESS FOLLOWING ITS CALL
                                ;//////////////////////////////////////////////////////////////////////
033714  017637  000000  001172  HAD:    MOV     @(SP),$TMP0     ;GET ADDRESS TO BE USED
033722  062716  000002                  ADD     #2,(SP)         ;ADJUST PC
033726  062737  002000  001172          ADD     #2000,$TMP0     ;CALC. ADDR WITH ADDRESS BIT A10 COMPLEMENTED
033734  042737  174000  001172          BIC     #174000,$TMP0   ;MASK A15-A11
033742  032737  002000  001172          BIT     #2000,$TMP0     ;BIT 10 SET?
033750  001004                          BNE     1$              ;BRANCH IF YES
033752  062737  060000  001172          ADD     #BUFL,$TMP0     ;CALC TEST BUFFER ADDR.
033760  000406                          BR      2$

033762  042737  002000  001172  1$:     BIC     #2000,$TMP0     ;ADJUST ADDRESS BIT A10
033770  062737  062000  001172          ADD     #BUFH,$TMP0     ;CALC TEST BUFFER ADR.
033776  000207                  2$:     RTS     PC              ;RETURN
                                ;//////////////////////////////////////////////////////////////////////
                                        ;SUBROUTINE TO SEE IF A NEW UNIBUS EXER. IS USED AND TO SETUP AN
                                        ;RTI IN ITS INTERRUPT VECTOR
                                ;//////////////////////////////////////////////////////////////////////

034000  023727  001232  034046  VEC:    CMP     SETUP,#NUBEN    ;NEW UBE USED?
034006  001016                          BNE     1$              ;BRANCH IF NO
034010  013737  001226  001172          MOV     IVEC,$TMP0      ;GET ITS INTERRUPT VECTOR
034016  062737  000002  001172          ADD     #2,$TMP0
034024  013777  001172  145174          MOV     $TMP0,@IVEC     ;PUT ON RTI
034032  012777  000002  145132          MOV     #RTI,@$TMP0     ;IN ITS INTERRUPT AREA
034040  005037  177776                  CLR     @#PSW           ;LOWER PRIORITY LEVEL FOR INTERRUPTS
034044  000207                  1$:     RTS     PC              ;RETURN

                                ;//////////////////////////////////////////////////////////////////////
                                        ;SUBROUTINE TO SETUP THE NEW UNIBUS EXERCISOR TO DO ONE NPR DATO
                                        ;TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL
                                ;//////////////////////////////////////////////////////////////////////

034046  005037  001204          NUBEN:  CLR     $TMP5           ;INIT COUNTER TO WAIT FOR RDY BIT
034052  105777  145134          2$:     TSTB    @CREG1          ;READY BIT SET?
034056  100421                          BMI     1$              ;BRANCH IF YES
034060  005237  001204                  INC     $TMP5           ;WAIT FOR RDY TO SET
034064  001372                          BNE     2$              ;BRANCH IF HAVEN'T WAITED MAX TIME
```

```
034066  032777  020000  145040                 BIT     #SW13,@SWR       ;INHIBIT TYPEOUTS?
034074  001004                                 BNE     3$               ;BRANCH IF YES
034076  104401  041645                         TYPE    ,MSG13           ;DEVICE READY BIT DOES NOT SET
034102  104401  041705                         TYPE    ,MSG14           ;FURTHER NPR DEVICE TESTS ABORTED
034106  005726                         3$:     TST     (SP)+            ;RESTORE STACK FROM SUBROUTINE CALL
034110  042737  000001  177572                 BIC     #1,@#MMR0        ;KT OFF IF ON
034116  000137  033020                         JMP     $EOP             ;GO TO END OF PROGRAM
034122  005077  145070                 1$:     CLR     @CREG3           ;CLEAR ANY ERROR BITS SET
034126  012777  003040  145056                 MOV     #3040,@CREG1     ;HAVE UBE DO INPR DATO DATA XFER
034134  005077  145054                         CLR     @CREG2           ;HAVE UBE DO INPR DATO DATA XFER
034140  012777  177777  145054                 MOV     #177777,@CREG5   ;CYCLE COUNT=1 XFER
034146  017677  000000  145044                 MOV     @(SP),@CREG4     ;GET ADDRESS FOR XFER
034154  062716  000002                         ADD     #2,(SP)          ;GET HIGH ADDRESS BITS A17, A16
034160  057677  000000  145026                 BIS     @(SP),@CREG2     ;PUT ADDRESS BITS IN CONTROL REG
034166  062716  000002                         ADD     #2,(SP)          ;ADJUST PC FOR RETURN
034172  000207                                 RTS     PC               ;RETURN

                                       ;//////////////////////////////////////////////////////////////////////
                                               ;SUBROUTINE TO SETUP THE OLD UNIBUS EXERCISOR TO DO 1 NPR DATO
                                               ;TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL.
                                       ;//////////////////////////////////////////////////////////////////////

034174  012737  050200  170006         HUBFO:  MOV     #50200,@#170006  ;HAVE UBE DO 1 NPR DATO AND RELEASE BUS
034202  012737  000002  170004                 MOV     #2,@#170004      ;SET BYTE COUNT FOR 1 WORD XFER
034210  017637  000000  170002                 MOV     @(SP),@#170002   ;SET UP XFER ADDRESS
034216  062716  000004                         ADD     #4,(SP)          ;ADJUST PC FOR RETURN
034222  000207                                 RTS     PC               ;RETURN
034224  000000                         FAKE:   .WORD   0                ;FAKE ERROR REG. MSB=0 FOR NO ERRORS

                                       ;//////////////////////////////////////////////////////////////////////
                                               ;SUBROUTINE TO SETUP AN RK05 FOR 1 NPR DATO
                                               ;TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL
                                       ;//////////////////////////////////////////////////////////////////////

034226  005037  001204                 HRK05:  CLR     $TMP5            ;INIT COUNTER TO WAIT FOR RDY BIT
034232  105737  177404                 2$:     TSTB    @#RKCS           ;IS CONTROLLER RDY?
034236  100421                                 BMI     1$               ;BRANCH IF YES
034240  005237  001204                         INC     $TMP5            ;WAIT FOR RDY TO SET
034244  001372                                 BNE     2$               ;BRANCH IF HAVEN'T WAITED MAX TIME
034246  042737  000001  177572         5$:     BIC     #1,@#MMR0        ;KT OFF IF ON
034254  032777  020000  144652                 BIT     #SW13,@SWR       ;INHIBIT TYPEOUTS?
034262  001004                                 BNE     9$               ;BRANCH IF YES
034264  104401  041645                         TYPE    ,MSG13           ;DEVICE RDY BIT DOES NOT SET
034270  104401  041705                         TYPE    ,MSG14           ;FURTHER NPR TESTS ABORTED
034274  005726                         9$:     TST     (SP)+            ;RESTORE STACK FROM SUBROUTINE CALL
034276  000137  033020                         JMP     $EOP             ;GO TO END OF PROGRAM

034302  005037  001204                 1$:     CLR     $TMP5            ;INIT COUNTER TO WAIT FOR RDY BIT
034306  032737  000100  177400         4$:     BIT     #100,@#RKDS      ;IS DRIVE RDY?
034314  001004                                 BNE     3$               ;BRANCH IF YES
034316  005237  001204                         INC     $TMP5            ;WAIT FOR RDY TO SET
034322  001371                                 BNE     4$               ;BRANCH IF HAVEN'T WAITED MAX TIME
034324  000750                                 BR      5$               ;REPORT DEVICE NOT READY

034326  012737  000001  177404         3$:     MOV     #1,@#RKCS        ;RESET CONTROLLER
034334  005037  001204                         CLR     $TMP5            ;INIT COUNTER TO WAIT FOR RDY BIT
```

```
034340  105737  177404                 7$:     TSTB    @#RKCS           ;CONTROLLER RDY?
034344  100404                                 BMI     6$               ;BRANCH IF YES
034346  005237  001204                         INC     $TMP5            ;WAIT FOR RDY TO SET
034352  001372                                 BNE     7$               ;BRANCH IF HAVEN'T WAITED MAX TIME
034354  000734                                 BR      5$               ;REPORT DEVICE NOT RDY

034356  012737  177777  177406         6$:     MOV     #-1,@#RKWC       ;SET WORD COUNT FOR 1 XFER
034364  013737  001214  177412                 MOV     @#CREG2,@#RKDA   ;SET UP DISK ADDRESS REG
034372  012737  000004  177404                 MOV     #4,@#RKCS        ;SET UP DISK TO DO DATO
034400  017637  000000  177410                 MOV     @(SP),@#RKBA     ;SET UP XFER ADDRESS
034406  062716  000002                         ADD     #2,(SP)          ;LOOK AT HIGH ADDRESS BITS
034412  017637  000000  001204                 MOV     @(SP),$TMP5      ;GET HIGH ADDRESS BITS
034420  062716  000002                         ADD     #2,(SP)          ;ADJUST PC FOR RETURN
034424  005037  001202                         CLR     $TMP4            ;INIT COUNT FOR SHIFT
034430  006337  001204                 8$:     ASL     $TMP5            ;SHIFT ADDRESS BITS TO RKCS ADDR, BIT'S POSITION
034434  005237  001202                         INC     $TMP4            ;COUNT SHIFTS
034440  023727  001202  000004                 CMP     $TMP4,#4         ;ALL CONE?
034446  001370                                 BNE     8$               ;BRANCH IF NO
034450  053737  001204  177404                 BIS     $TMP5,@#RKCS     ;SET UP THE EXTENDED MEMORY BITS
034456  000207                                 RTS     PC               ;RETURN

                                       ;//////////////////////////////////////////////////////////////////////
                                               ;SUBROUTINE TO SETUP AN RP03 TO DO 1 NPR DATO
                                               ;TO THE ADDRESS FOLLOWING THE SUBROUTINE CALL
                                       ;//////////////////////////////////////////////////////////////////////

034460  005737  176714                 HRP03:  TST     @#RPCS           ;ANY ERRORS?
034464  001416                                 BEQ     1$               ;BRANCH IF NO
034466  042737  000001  177572                 BIC     #1,@#MMR0        ;KT OFF IF ON
034474  032777  020000  144432                 BIT     #SW13,@SWR       ;INHIBIT TYPEOUTS?
034502  001004                                 BNE     2$               ;BRANCH IF YES
034504  104401  041614                         TYPE    ,MSG12           ;DEVICE ERROR BIT SET
034510  104401  041705                         TYPE    ,MSG14           ;FURTHER NPR TESTS ABORTED
034514  005726                         2$:     TST     (SP)+            ;RESTORE STACK FROM SUBROUTINE CALL
034516  000137  033020                         JMP     $EOP             ;GO TO END OF PROG

034522  005037  001204                 1$:     CLR     $TMP5            ;INIT COUNTER TO WAIT FOR RDY BIT
034526  105737  176714                 4$:     TSTB    @#RPCS           ;CONTROLLER RDY?
034532  100421                                 BMI     3$               ;BRANCH IF YES
034534  005237  001204                         INC     $TMP5            ;WAIT FOR RDY TO SET
034540  001372                                 BNE     4$               ;BRANCH IF HAVEN'T WAITED MAX TIME
034542  042737  000001  177572         8$:     BIC     #1,@#MMR0        ;KT OFF IF ON
034550  032777  020000  144356                 BIT     #SW13,@SWR       ;INHIBIT TYPEOUTS?
034556  001004                                 BNE     5$               ;BRANCH IF YES
034560  104401  041645                         TYPE    ,MSG13           ;DEVICE RDY BIT DID NOT SET
034564  104401  041705                         TYPE    ,MSG14           ;FURTHER NPR DEVICE TEST ABORTED
034570  005726                         5$:     TST     (SP)+            ;RESTORE STACK FROM SUBROUTINE CALL
034572  000137  033020                         JMP     $EOP             ;GO TO END OF PROG

034576  005037  001204                 3$:     CLR     $TMP5            ;INIT COUNTER TO WAIT FOR RDY BIT
034602  005737  176710                 7$:     TST     @#RPDS           ;IS DEVICE RDY?
034606  100404                                 BMI     6$               ;BRANCH IF YES
034610  005237  001204                         INC     $TMP5            ;WAIT FOR RDY TO SET
034614  001372                                 BNE     7$               ;BRANCH IF HAVEN'T WAITED MAX TIME
034616  000751                                 BR      8$               ;REPORT RDY DID NOT SET

```

```
034620  012737  177776  176716  6$:     MOV     #-2,@#RPWC       ;SET UP TO DO MIN # OF XFERS(2)
034626  013737  001214  176714          MOV     CREG2,@#RPCS     ;START TO SETUP CONTROLLER FOR NPR DATO
034634  017637  000000  176720          MOV     @(SP),@#RPBA     ;SETUP XFER ADDRESS
034642  062716  000002                  ADD     #2,(SP)          ;LOOK AT HIGH XFER ADDRESS
034646  017637  000000  001202          MOV     @(SP),$TMP4      ;GET HIGH XFER ADDR.
034654  062716  000002                  ADD     #2,(SP)          ;ADJUST PC FOR RETURN
034660  005037  001204                  CLR     $TMP5            ;INIT SHIFT COUNTER
034664  006337  001202          9$:     ASL     $TMP4            ;SHIFT ADDR. BITS TO COINCIDE WITH RPCS EXTENDED ADDR. B
034670  005237  001204                  INC     $TMP5            ;COUNT SHIFTS
034674  022737  000004  001204          CMP     #4,$TMP5         ;FINISHED?
034702  001370                          BNE     9$               ;BRANCH IF NO
034704  053737  001202  176714          BIS     $TMP4,@#RPCS     ;SETUP THE EXTENDED MEM ADDR.
034712  000207                          RTS     PC               ;RETURN

                                        ;/////////////////////////////////////////////////////////////////////
                                                ;SUBROUTINE TO SETUP A TU10 TO DO NPR DATO XFERS
                                                ;TO THE STARTING ADDRESS FOLLOWING THE SUBROUTINE CALL
                                        ;/////////////////////////////////////////////////////////////////////

034714  052737  010000  172522  HTU10:  BIS     #10000,@#MTC     ;POWER CLEAR THE UNIT
034722  000240                          NOP                      ;WAIT FOR POWER CLEAR
034724  000240                          NOP
034726  012737  177777  172524          MOV     #-1,@#MTBRC      ;PREPARE TO BACKSPACE ONE RECORD
034734  013737  001214  172522          MOV     CREG2,@#MTC      ;GET CONTROL MASK
034742  052737  000012  172522          BIS     #12,@#MTC        ;SET UP BACKSPACE COMMAND
034750  005237  172522                  INC     @#MTC            ;BACKSPACE
034754  005037  001204                  CLR     $TMP5            ;INIT COUNTER TO WAIT FOR RDY
034760  032737  000001  172520  2$:     BIT     #1,@#MTS         ;UNIT DONE?
034766  001021                          BNE     1$               ;BRANCH IF YES
034770  005237  001204                  INC     $TMP5            ;WAIT TILL UNIT DONE
034774  001371                          BNE     2$               ;BRANCH IF HAVEN'T WAITED MAX TIME
034776  005726                          TST     (SP)+            ;RESTORE STACK FROM SUBROUTINE CALL
035000  042737  000001  177572          BIC     #1,@#MMR0        ;KT OFF IF ON
035006  032777  020000  144120          BIT     #SW13,@SWR       ;INHIBIT TYPEOUTS?
035014  001004                          BNE     3$               ;BRANCH IF YES
035016  104401  041645                  TYPE    ,MSG13           ;RDY BIT DID NOT SET
035022  104401  041705                  TYPE    ,MSG14           ;ABORT ALL TESTS USING NPR DEVICE
035026  000137  033020          3$:     JMP     $EOP             ;GO TO END OF PROGRAM

035032  013737  001214  172522  1$:     MOV     CREG2,@#MTC      ;GET CONTROL MASK
035040  052737  000002  172522          BIS     #2,@#MTC         ;SETUP CONTROL TO DO READ
035046  012737  177760  172524          MOV     #-20,@#MTBRC     ;PREPARE TO READ MIN # OF BYTES (20(8))
035054  017637  000000  172526          MOV     @(SP),@#MTCMA    ;SETUP XFER ADDRESS
035062  062716  000002                  ADD     #2,(SP)          ;LOOK AT HIGH ADDRESS
035066  017637  000000  001202          MOV     @(SP),$TMP4      ;GET HIGH ADDRESS
035074  005037  001204                  CLR     $TMP5            ;INIT SHIFT COUNTER
035100  006337  001202          4$:     ASL     $TMP4            ;SHIFT ADDR. BITS TO COINCIDE WITH MTC ADDR BITS
035104  005237  001204                  INC     $TMP5            ;COUNT SHIFTS
035110  022737  000004  001204          CMP     #4,$TMP5         ;DONE?
035116  001370                          BNE     4$               ;BRANCH IF NO
035120  053737  001202  172522          BIS     $TMP4,@#MTC      ;SETUP HIGH ADDRESS BITS
035126  062716  000002                  ADD     #2,(SP)          ;ADJUST PC FOR RETURN
035132  000207                          RTS     PC               ;RETURN
                                        ;/////////////////////////////////////////////////////////////////////
                                                ;SUBROUTINE TO RID CACHE OF BAD PARITY BY
                                                ;OVERWRITTING IT WHEN CACHE IS OFF
```

```
                                        ;/////////////////////////////////////////////////////////////////////
035134  012705  060000          SWEEP:  MOV     #BUFL,R5         ;GET STARTING ADDRESS
035140  011525                  64$:    MOV     (R5),(R5)+       ;WRITE ALL CACHE WITH GOOD PARITY
035142  020527  064000                  CMP     R5,#BUFL+4000    ;ALL CACHE WRITTEN?
035146  001374                          BNE     64$              ;BRANCH IF NO
035150  000207                          RTS     PC               ;RETURN

                                        .SBTTL  SCOPE HANDLER ROUTINE

                                        ;;*************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS, IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW11=1         INHIBIT ITERATIONS
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT

035152                                  $SCOPE:
035152  032777  040000  143754  1$:     BIT     #BIT14,@SWR      ;;LOOP ON PRESENT TEST?
035160  001104                          BNE     $OVER            ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
035162  000416                  $XTSTR: BR      6$               ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                 ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
035164  013746  000004                  MOV     @#ERRVEC,-(SP)   ;;SAVE THE CONTENTS OF THE ERROR VECTOR
035170  012737  035210  000004          MOV     #5$,@#ERRVEC     ;;SET FOR TIMEOUT
035176  005737  177060                  TST     @#177060         ;;TIME OUT ON XOR?
035202  012637  000004                  MOV     (SP)+,@#ERRVEC   ;;RESTORE THE ERROR VECTOR
035206  000453                          BR      $SVLAD           ;;GO TO THE NEXT TEST
035210  022626                  5$:     CMP     (SP)+,(SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
035212  012637  000004                  MOV     (SP)+,@#ERRVEC   ;;RESTORE THE ERROR VECTOR
035216  000413                          BR      7$               ;;LOOP ON THE PRESENT TEST
035220                          6$:     ;#####END OF CODE FOR THE XOR TESTER#####
035220  105737  001103          2$:     TSTB    $ERFLG           ;;HAS AN ERROR OCCURRED?
035224  001421                          BEQ     3$               ;;BR IF NO
035226  123737  001115  001103          CMPB    $ERMAX,$ERFLG    ;;MAX. ERRORS FOR THIS TEST OCCURRED?
035234  101015                          BHI     3$               ;;BR IF NO
035236  032777  001000  143670          BIT     #BIT09,@SWR      ;;LOOP ON ERROR?
035244  001404                          BEQ     4$               ;;BR IF NO
035246  013737  001110  001106  7$:     MOV     $LPERR,$LPADR    ;;SET LOOP ADDRESS TO LAST SCOPE
035254  000446                          BR      $OVER
035256  105037  001103          4$:     CLRB    $ERFLG           ;;ZERO THE ERROR FLAG
035262  005037  035406                  CLR     $TIMES           ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
035266  000415                          BR      1$               ;;ESCAPE TO THE NEXT TEST
035270  032777  004000  143636  3$:     BIT     #BIT11,@SWR      ;;INHIBIT ITERATIONS?
035276  001011                          BNE     1$               ;;BR IF YES
035300  005737  001244                  TST     $PASS            ;;IF FIRST PASS OF PROGRAM
035304  001406                          BEQ     1$               ;;      INHIBIT ITERATIONS
035306  005237  001104                  INC     $ICNT            ;;INCREMENT ITERATION COUNT
035312  023737  035406  001104          CMP     $TIMES,$ICNT     ;;CHECK THE NUMBER OF ITERATIONS MADE
035320  002024                          BGE     $OVER            ;;BR IF MORE ITERATION REQUIRED
035322  012737  000001  001104  1$:     MOV     #1,$ICNT         ;;REINITIALIZE THE ITERATION COUNTER
035330  013737  035410  035406          MOV     $MXCNT,$TIMES    ;;SET NUMBER OF ITERATIONS TO DO
035336  105237  001102          $SVLAD: INCB    $TSTNM           ;;COUNT TEST NUMBERS
```

```
035342  113737  001102  001247          MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
035350  011637  001106                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
035354  011637  001110                  MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
035360  005037  035606                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
035364  112737  000001  001115          MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
035372  013777  001102  143536  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
035400  013716  001106                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
035404  000002                          RTI                       ;;FIXES PS
035406  000000                  $TIMES: 0                         ;;NUMBER OF ITERATIONS TO PERFORM
035410  000005                  $MXCNT: 5.                        ;;MAX. NUMBER OF ITERATIONS
                                .SBTTL  ERROR HANDLER ROUTINE

                                ;;*************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

035412                          $ERROR:
035412  105237  001103          7$:     INCB    $ERFLG            ;;SET THE ERROR FLAG
035416  001775                          BEQ     7$                ;;DON'T LET THE FLAG GO TO ZERO
035420  013777  001102  143510          MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER AND ERROR FLAG
035426  032777  002000  143500          BIT     #BIT10,@SWR       ;;BELL ON ERROR?
035434  001402                          BEQ     1$                ;;NO - SKIP
035436  104401  035610                  TYPE    ,$BELL            ;;RING BELL
035442  005237  001112          1$:     INC     $ERTTL            ;;COUNT THE NUMBER OF ERRORS
035446  011637  001116                  MOV     (SP),$ERRPC       ;;GET ADDRESS OF ERROR INSTRUCTION
035452  162737  000002  001116          SUB     #2,$ERRPC
035460  117737  143432  001114          MOVB    @$ERRPC,$ITEMB    ;;STRIP AND SAVE THE ERROR ITEM CODE
035466  032777  020000  143440          BIT     #BIT13,@SWR       ;;SKIP TYPEOUT IF SET
035474  001004                          BNE     20$               ;;SKIP TYPEOUTS
035476  004737  035614                  JSR     PC,$ERRTYP        ;;GO TO USER ERROR ROUTINE
035502  104401  001207                  TYPE    ,$CRLF
035506                          20$:
035506  122737  000001  001256          CMPB    $APTENV,$ENV      ;;RUNNING IN APT MODE
035514  001007                          BNE     2$                ;;NO,SKIP APT ERROR REPORT
035516  113737  001114  035530          MOVB    $ITEMB,21$        ;;SET ITEM NUMBER AS ERROR NUMBER
035524  004737  037050                  JSR     PC,$ATY4          ;;REPORT FATAL ERROR TO APT
035530     000                  21$:    .BYTE   0
035531     000                          .BYTE   0
035532  000777                  22$:    BR      22$               ;;APT ERROR LOOP
035534  005777  143374          2$:     TST     @SWR              ;;HALT ON ERROR
035540  100001                          BPL     3$                ;;SKIP IF CONTINUE
035542  000000                          HALT                      ;;HALT ON ERROR!
035544  032777  001000  143362  3$:     BIT     #BIT09,@SWR       ;;LOOP ON ERROR SWITCH SET?
035552  001402                          BEQ     4$                ;;BR IF NO
035554  013716  001110                  MOV     $LPERR,(SP)       ;;FUDGE RETURN FOR LOOPING
035560  005737  035606          4$:     TST     $ESCAPE           ;;CHECK FOR AN ESCAPE ADDRESS
035564  001402                          BEQ     5$                ;;BR IF NONE
035566  013716  035606                  MOV     $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
```

```
035572                          5$:
035572  022737  033106  000042          CMP     #$ENDAD,@#42      ;;ACT-11 AUTO-ACCEPT?
035600  001001                          BNE     6$                ;;BRANCH IF NO
035602  000000                          HALT                      ;;YES
035604                          6$:
035604  000002                          RTI                       ;;RETURN
035606  000000                  $ESCAPE: .WORD  0                 ;;ESCAPE ON ERROR ADDRESS
035610  177607  000377          $BELL:  .ASCIZ  <207><377><377>   ;;ASCII CODE FOR BELL
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                ;;*************************************************************
                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

035614                          $ERRTYP:

035614  104401  001207                  TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
035620  010046                          MOV     R0,-(SP)          ;;SAVE R0
035622  005000                          CLR     R0                ;;PICKUP THE ITEM INDEX
035624  153700  001114                  BISB    @#$ITEMB,R0
035630  001004                          BNE     1$                ;;IF ITEM NUMBER IS ZERO, JUST
                                                                  ;;TYPE THE PC OF THE ERROR
035632  013746  001116                  MOV     $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT
                                                                  ;;ERROR ADDRESS
035636  104402                          TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
035640  000426                          BR      6$                ;;GET OUT
035642  005300                  1$:     DEC     R0                ;;ADJUST THE INDEX SO THAT IT WILL
035644  006300                          ASL     R0                ;;       WORK FOR THE ERROR TABLE
035646  006300                          ASL     R0
035650  006300                          ASL     R0
035652  062700  055074                  ADD     #$ERRTB,R0        ;;FORM TABLE POINTER
035656  012037  035666                  MOV     (R0)+,2$          ;;PICKUP "ERROR MESSAGE" POINTER
035662  001404                          BEQ     3$                ;;SKIP TYPEOUT IF NO POINTER
035664  104401                          TYPE                      ;;TYPE THE "ERROR MESSAGE"
035666  000000                  2$:     .WORD   0                 ;;"ERROR MESSAGE" POINTER GOES HERE
035670  104401  001207                  TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
035674  012037  035704          3$:     MOV     (R0)+,4$          ;;PICKUP "DATA HEADER" POINTER
035700  001404                          BEQ     5$                ;;SKIP TYPEOUT IF 0
035702  104401                          TYPE                      ;;TYPE THE "DATA HEADER"
035704  000000                  4$:     .WORD   0                 ;;"DATA HEADER" POINTER GOES HERE
035706  104401  001207                  TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
035712  011000                  5$:     MOV     (R0),R0           ;;PICKUP "DATA TABLE" POINTER
035714  001004                          BNE     7$                ;;GO TYPE THE DATA
035716  012600                  6$:     MOV     (SP)+,R0          ;;RESTORE R0
035720  104401  001207                  TYPE    ,$CRLF            ;;"CARRIAGE RETURN" & "LINE FEED"
035724  000207                          RTS     PC                ;;RETURN
035726                          7$:
035726  013046                          MOV     @(R0)+,-(SP)      ;;SAVE @(R0)+ FOR TYPEOUT
035730  104402                          TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
035732  005710                          TST     (R0)              ;;IS THERE ANOTHER NUMBER?
035734  001770                          BEQ     6$                ;;BR IF NO
035736  104401  035744                  TYPE    ,8$               ;;TYPE TWO(2) SPACES
035742  000771                          BR      7$                ;;LOOP
035744  020040     000          8$:     .ASCIZ  /  /              ;;TWO(2) SPACES
        035750                          .EVEN
                                        .SBTTL          ROUTINE TO SIZE MEMORY
                                        ;;******************************************************************
                                        ;*CALL:
                                        ;*       JSR             PC,$SIZE
                                        ;*       RETURN
                                        ;*$LSTAD WILL CONTAIN:
                                        ;*       WITH KT11--LAST VIRTUAL ADDRESS OF THE LAST BANK
                                        ;*       WITHOUT KT11 --LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                        ;*$LSTBK  WILL CONTAIN THE LAST BANK AS A BAF
                                        ;*
                                        ;*$KT11 IS THE MEMORY MANAGEMENT KEY
                                        ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                        ;*       MUST BE SET UP BEFORE THE CALL
                                        ;*BIT15 =0 DON'T HAVE MEMORY MANAGEMENT OPTION
                                        ;*       DETERMINED BY ROUTINE

 035750  010046                         $SIZE:  MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
 035752  010146                                 MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
 035754  010246                                 MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
 035756  010346                                 MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
 035760  013746  000004                         MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
 035764  013746  000006                         MOV     @#ERRVEC+2,-(SP)
 035770  010600                                 MOV     SP,R0           ;;SAVE THE STACK POINTER
                                        ;;SET THE ERRVEC PS TO THE PRESENT PS
 035772  013746  000034                         MOV     @#TRAPVEC,-(SP) ;;SAVE CURRENT TRAP VECTOR
 035776  012737  036006  000034                 MOV     #64$,@#TRAPVEC  ;;SETUP NEW TRAP VECTOR
 036004  104400                                 TRAP                    ;;PUSH OLD PSW AND PC ON STACK
 036006  016637  000002  000006 64$:    MOV     2(SP),@#ERRVEC+2  ;;SAVE PSW IN @#ERRVEC+2
 036014  012716  036022                         MOV     #65$,(SP)       ;;REPLACE OLD PC WITH NEW
 036020  000002                                 RTI                     ;;RESTORE PSW
 036022  012637  000034                 65$:    MOV     (SP)+,@#TRAPVEC ;;RESTORE OLD TRAP VECTOR
 036026  012701  003776                         MOV     #3776,R1        ;;SETUP ADDRESS
 036032  105727                                 TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
 036034  000200                         $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
 036036  100062                                 BPL     $CORE           ;;BR IF NO
 036040  012737  036176  000004                 MOV     #$KTNEX,@#ERRVEC  ;;SET FOR TIMEOUT
 036046  005737  177572                         TST     @#SR0           ;;KT11 ARE YOU THERE?
 036052  052737  100000  036034                 BIS     #100000,$KT11   ;;YES--SET KT11 KEY
 036060  005046                                 CLR     -(SP)           ;INITIALIZE FOR "PAR" LOADING
 036062  012702  172340                         MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
 036066  012703  000010                         MOV     #^D8,R3         ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
 036072  012762  077406  177740 1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
 036100  011622                                 MOV     (SP),(R2)+      ;;LOAD "PAR"
 036102  062716  000200                         ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
 036106  077307                                 SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
 036110  012742  177600                         MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
 036114  005042                                 CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
 036116  012737  036134  000004                 MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
 036124  012737  000020  172516                 MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
 036132  000401                                 BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
 036134  022626                         2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
 036136  005237  177572                 3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
 036142  012737  036166  000004                 MOV     #$KTOUT,@#ERRVEC  ;;SET FOR TIME OUT
 036150  005737  143776                 4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
 036154  062712  000040                         ADD     #40,(R2)        ;;MAKE A 1K STEP
 036160  023712  172356                         CMP     @#KIPAR7,(R2)   ;;LAST ONE?
```

```
 036164  101371                                 BHI     4$              ;;NO--TRY IT
 036166  011202                         $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
 036170  005037  177572                         CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
 036174  000421                                 BR      $SIZEX
 036176  042737  100000  036034 $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
 036204  012737  036234  000004 $CORE:  MOV     #$CROUT,@#ERRVEC  ;;SET FOR TIMEOUT
 036212  005002                                 CLR     R2              ;;SET UP BANK
 036214  062701  004000                 1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
 036220  062702  000040                         ADD     #40,R2          ;;1K STEP
 036224  005711                                 TST     (R1)            ;;TRAP ON TIME OUT
 036226  022701  177776                         CMP     #177776,R1      ;;LAST ONE
 036232  001370                                 BNE     1$              ;;NO--TRY AGAIN
 036234  162701  004000                 $CROUT: SUB     #4000,R1
 036240  162702  000040                 $SIZEX: SUB     #40,R2          ;;DROP BACK
 036244  010006                                 MOV     R0,SP           ;;RESTORE THE STACK
 036246  012637  000006                         MOV     (SP)+,@#ERRVEC+2  ;;RESTORE ERROR VECTOR
 036252  012637  000004                         MOV     (SP)+,@#ERRVEC
 036256  010137  036320                         MOV     R1,$LSTAD       ;;LAST ADDRESS
 036262  010237  036322                         MOV     R2,$LSTBK       ;;LAST BANK
 036266  012603                                 MOV     (SP)+,R3        ;;RESTORE R3
 036270  012602                                 MOV     (SP)+,R2        ;;RESTORE R2
 036272  012601                                 MOV     (SP)+,R1        ;;RESTORE R1
 036274  012600                                 MOV     (SP)+,R0        ;;RESTORE R0

 036276  010046                                 MOV     R0,-(SP)        ;SAVE R0 FOR MED INST
 036300  076600                                 MED                     ;GET CONTENTS OF LOG REG
 036302  000022                                 .WORD   RLOG
 036304  052700  100001                         BIS     #100001,R0      ;ENABLE ERROR LOG & LOG FIRST MODE
 036310  076600                                 MED                     ;UNLOCK ERROR LOG
 036312  000222                                 .WORD   WLOG
 036314  012600                                 MOV     (SP)+,R0        ;RESTORE R0

 036316  000207                                 RTS     PC
 036320  000000                         $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
 036322  000000                         $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
                                        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

 036324                                 $TYPDS:
 036324  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 036326  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 036330  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 036332  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 036334  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 036336  012746  020200                         MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 036342  016605  000020                         MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 036346  100004                                 BPL     1$              ;;BR IF INPUT IS POS.
```

```
036350  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
036352  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
036360  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
036362  012703  036540                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
036366  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
036372  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
036374  016001  036530                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
036400  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
036402  002402                          BLT     4$              ;;BR IF DONE
036404  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
036406  000774                          BR      3$
036410  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
036412  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
036414  001002                          BNE     5$              ;;FALL THROUGH IF 0
036416  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
036420  100407                          BMI     7$              ;;BR IF YES
036422  106316                  5$:     ASLB    (SP)            ;;MSD?
036424  103003                          BCC     6$              ;;BR IF NO
036426  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
036434  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
036440  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
036444  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
036446  005720                          TST     (R0)+           ;;JUST INCREMENTING
036450  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
036454  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
036456  003002                          BGT     8$              ;;GO TO EXIT
036460  010502                          MOV     R5,R2           ;;GET THE LSD
036462  000764                          BR      6$              ;;GO CHANGE TO ASCII
036464  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
036466  100003                          BPL     9$              ;;BR IF NO
036470  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
036476  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
036500  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
036502  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
036504  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
036506  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
036510  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
036512  104401  036540                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
036516  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
036524  012616                          MOV     (SP)+,(SP)
036526  000002                          RTI                     ;;RETURN TO USER
036530  023420                  $DTBL:  10000.
036532  001750                          1000.
036534  000144                          100.
036536  000012                          10.
036540  000004                  $DBLK:  .BLKW   4
                                .SBTTL  TYPE ROUTINE

                                ;;*************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE, MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
```

```
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

036550  105737  001153          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
036554  100002                          BPL     1$              ;;BR IF YES
036556  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
036560  000430                          BR      3$              ;;LEAVE
036562  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
036564  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
036570  122737  000001  001256          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
036576  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
036600  132737  000100  001257          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
036606  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
036610  010037  036620                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
036614  004737  037040                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
036620  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
036622  132737  000040  001257  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
036630  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
036632  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
036634  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
036636  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
036640  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
036642  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
036646  000002                          RTI                     ;;RETURN
036650  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
036654  001430                          BEQ     8$
036656  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
036662  001006                          BNE     5$
036664  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
036666  104401                          TYPE                    ;;TYPE A CR AND LF
036670  001207                          $CRLF
036672  105037  037026                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
036676  000755                          BR      2$              ;;GET NEXT CHARACTER
036700  004737  036762          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
036704  123726  001152          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
036710  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
036712  013746  001150                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
036716  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
036722  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
036724  004737  036762                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
036730  105337  037026                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
036734  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

036736  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
036742  004737  036762          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
036746  132737  000007  037026          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
036754  001372                          BNE     9$              ;;TAB STOP
036756  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
036760  000724                          BR      2$              ;;GET NEXT CHARACTER
```

```
037762  105777  142156          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
036766  100375                          BPL     $TYPEC
036770  116677  000002  142150          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
036776  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
037004  001003                          BNE     1$              ;;BRANCH IF NO
037006  105037  037026                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
037012  000406                          BR      $TYPEX          ;;EXIT
037014  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
037022  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
037024  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
037026  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
037030  000207                  $TYPEX: RTS     PC

                                .SBTTL  APT COMMUNICATIONS ROUTINE


                                ;;*************************************************************
037032  112737  000001  037276  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
037040  112737  000001  037274  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
037046  000403                          BR      $ATYC
037050  112737  000001  037276  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
037056                          $ATYC:
037056  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
037060  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
037062  105737  037274                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
037066  001450                          BEQ     5$              ;;IF NOT: BR
037070  122737  000001  001256          CMPB    $APTENV,$ENV    ;;OPERATING UNDER APT?
037076  001031                          BNE     3$              ;;IF NOT: BR
037100  132737  000100  001257          BITB    $APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
037106  001425                          BEQ     3$              ;;IF NOT: BR
037110  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
037114  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
037122  005737  001236          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
037126  001375                          BNE     1$              ;;IF NOT: WAIT
037130  010037  001252                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
037134  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
037136  001376                          BNE     2$
037140  163700  001252                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
037144  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
037146  010037  001254                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
037152  012737  000004  001236          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
037160  000413                          BR      5$
037162  017637  000004  037206  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
037170  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
037176  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
037202  004737  036550                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
037206  000000                  4$:     .WORD   0
037210                          5$:
037210  105737  037276          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
037214  001416                          BEQ     12$             ;;IF NOT: BR
037216  005737  001256                  TST     $ENV            ;;RUNNING UNDER APT?
037222  001413                          BEQ     12$             ;;IF NOT: BR
037224  005737  001236          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
037230  001375                          BNE     11$             ;;IF NOT: WAIT
037232  017637  000004  001240          MOV     @4(SP),$FATAL   ;;GET ERROR #
037240  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
037246  005237  001236                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
```

```
037252  105037  037276          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
037256  105037  037275                  CLRB    $LFLG           ;;CLEAR LOG FLAG
037262  105037  037274                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
037266  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
037270  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
037272  000207                          RTS     PC              ;;RETURN
037274     000                  $MFLG:  .BYTE   0               ;;MESSG. FLAG
037275     000                  $LFLG:  .BYTE   0               ;;LOG FLAG
037276     000                  $FFLG:  .BYTE   0               ;;FATAL FLAG
        037300                          .EVEN
        000200                  APTSIZE=200
        000001                  APTENV=001
        000100                  APTSPOOL=100
        000040                  APTCSUP=040
                                .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

037300  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
037304  116637  000001  037523          MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
037312  112637  037525                  MOVB    (SP)+,$0MODE+1  ;;NUMBER OF DIGITS TO TYPE
037316  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
037322  000406                          BR      $TYPON
037324  112737  000001  037523  $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
037332  112737  000006  037525          MOVB    #6,$0MODE+1     ;;SET FOR SIX(6) DIGITS
037340  112737  000005  037522  $TYPON: MOVB    #5,$0CNT        ;;SET THE ITERATION COUNT
037346  010346                          MOV     R3,-(SP)        ;;SAVE R3
037350  010446                          MOV     R4,-(SP)        ;;SAVE R4
037352  010546                          MOV     R5,-(SP)        ;;SAVE R5
037354  113704  037525                  MOVB    $0MODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
037360  005404                          NEG     R4
037362  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
037366  110437  037524                  MOVB    R4,$0MODE       ;;SAVE IT FOR USE
037372  113704  037523                  MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
037376  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
```

```
037402  005003                          CLR     R3                      ;;CLEAR THE OUTPUT WORD
037404  006105                  1$:     ROL     R5                      ;;ROTATE MSB INTO "C"
037406  000404                          BR      3$                      ;;GO DO MSB
037410  006105                  2$:     ROL     R5                      ;;FORM THIS DIGIT
037412  006105                          ROL     R5
037414  006105                          ROL     R5
037416  010503                          MOV     R5,R3
037420  006103                  3$:     ROL     R3                      ;;GET LSB OF THIS DIGIT
037422  105337  037524                  DECB    $OMODE                  ;;TYPE THIS DIGIT?
037426  100016                          BPL     7$                      ;;BR IF NO
037430  042703  177770                  BIC     #177770,R3              ;;GET RID OF JUNK
037434  001002                          BNE     4$                      ;;TEST FOR 0
037436  005704                          TST     R4                      ;;SUPPRESS THIS 0?
037440  001403                          BEQ     5$                      ;;BR IF YES
037442  005204                  4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
037444  052703  000060                  BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
037450  052703  000040          5$:     BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
037454  110337  037520                  MOVB    R3,8$                   ;;SAVE FOR TYPING
037460  104401  037520                  TYPE    ,8$                     ;;GO TYPE THIS DIGIT
037464  105337  037522          7$:     DECB    $OCNT                   ;;COUNT BY 1
037470  003347                          BGT     2$                      ;;BR IF MORE TO DO
037472  002402                          BLT     6$                      ;;BR IF DONE
037474  005204                          INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
037476  000744                          BR      2$                      ;;GO DO THE LAST DIGIT
037500  012605                  6$:     MOV     (SP)+,R5                ;;RESTORE R5
037502  012604                          MOV     (SP)+,R4                ;;RESTORE R4
037504  012603                          MOV     (SP)+,R3                ;;RESTORE R3
037506  016666  000002  000004          MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
037514  012616                          MOV     (SP)+,(SP)
037516  000002                          RTI                             ;;RETURN
037520     000                  8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
037521     000                          .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
037522     000                  $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
037523     000                  $0FILL: .BYTE   0                       ;;ZERO FILL SWITCH
037524  000000                  $OMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
                                .SBTTL  TTY INPUT ROUTINE


                                ;;*************************************************************
                                .ENABL  LSB

                                .DSABL  LSB


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

037526  011646                  $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC
037530  016666  000004  000002          MOV     4(SP),2(SP)             ;;SAVE THE PS
037536  105777  141376          1$:     TSTB    @$TKS                   ;;WAIT FOR
037542  100375                          BPL     1$                      ;;A CHARACTER
037544  117766  141372  000004          MOVB    @$TKB,4(SP)             ;;READ THE TTY
```

```
037552  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
037560  026627  000004  000023          CMP     4(SP),#23               ;;IS IT A CONTROL-S?
037566  001013                          BNE     3$                      ;;BRANCH IF NO
037570  105777  141344          2$:     TSTB    @$TKS                   ;;WAIT FOR A CHARACTER
037574  100375                          BPL     2$                      ;;LOOP UNTIL ITS THERE
037576  117746  141340                  MOVB    @$TKB,-(SP)             ;;GET CHARACTER
037602  042716  177600                  BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
037606  022627  000021                  CMP     (SP)+,#21               ;;IS IT A CONTROL-Q?
037612  001366                          BNE     2$                      ;;IF NOT DISCARD IT
037614  000750                          BR      1$                      ;;YES, RESUME
037616  026627  000004  000140  3$:     CMP     4(SP),#140              ;;IS IT UPPER CASE?
037624  002407                          BLT     4$                      ;;BRANCH IF YES
037626  026627  000004  000175          CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
037634  003003                          BGT     4$                      ;;BRANCH IF YES
037636  042766  000040  000004          BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
037644  000002                  4$:     RTI                             ;;GO BACK TO USER
                                ;;*************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

037646  010346                  $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
037650  005046                          CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
037652  012703  040102          1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
037656  022703  040112          2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
037662  101456                          BLOS    4$                      ;;BR IF YES
037664  104406                          RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
037666  112613                          MOVB    (SP)+,(R3)              ;;GET CHARACTER
037670  122713  000177          10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
037674  001022                          BNE     5$                      ;;BR IF NO
037676  005716                          TST     (SP)                    ;;IS THIS THE FIRST RUBOUT?
037700  001007                          BNE     6$                      ;;BR IF NO
037702  112737  000134  040100          MOVB    #'\,9$                  ;;TYPE A BACK SLASH
037710  104401  040100                  TYPE    ,9$
037714  012716  177777                  MOV     #-1,(SP)                ;;SET THE RUBOUT KEY
037720  005303                  6$:     DEC     R3                      ;;BACKUP BY ONE
037722  020327  040102                  CMP     R3,#$TTYIN              ;;STACK EMPTY?
037726  103434                          BLO     4$                      ;;BR IF YES
037730  111337  040100                  MOVB    (R3),9$                 ;;SETUP TO TYPEOUT THE DELETED CHAR.
037734  104401  040100                  TYPE    ,9$                     ;;GO TYPE
037740  000746                          BR      2$                      ;;GO READ ANOTHER CHAR.
037742  005716                  5$:     TST     (SP)                    ;;RUBOUT KEY SET?
037744  001406                          BEQ     7$                      ;;BR IF NO
037746  112737  000134  040100          MOVB    #'\,9$                  ;;TYPE A BACK SLASH
037754  104401  040100                  TYPE    ,9$
037760  005016                          CLR     (SP)                    ;;CLEAR THE RUBOUT KEY
037762  122713  000025          7$:     CMPB    #25,(R3)                ;;IS CHARACTER A CTRL U?
037766  001003                          BNE     8$                      ;;BR IF NO
037770  104401  040112                  TYPE    ,$CNTLU                 ;;TYPE A CONTROL "U"
037774  000726                          BR      1$                      ;;GO START OVER
037776  122713  000022          8$:     CMPB    #22,(R3)                ;;IS CHARACTER A "^R"?
040002  001011                          BNE     3$                      ;;BRANCH IF NO
040004  105013                          CLRB    (R3)                    ;;CLEAR THE CHARACTER
040006  104401  001207                  TYPE    ,$CRLF                  ;;TYPE A "CR" & "LF"
```

```
040012  104401  040102                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
040016  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
040020  104401  001206          4$:     TYPE    ,$QUES          ;;TYPE A "?"
040024  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
040026  111337  040100          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
040032  104401  040100                  TYPE    ,9$
040036  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
040042  001305                          BNE     2$              ;;LOOP IF NOT RETURN
040044  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
040050  104401  001210                  TYPE    ,$LF            ;;TYPE A LINE FEED
040054  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
040056  012603                          MOV     (SP)+,R3        ;;RESTORE R3
040060  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
040062  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
040070  012766  040102  000004          MOV     #$TTYIN,4(SP)
040076  000002                          RTI                     ;;RETURN
040100     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
040101     000                          .BYTE   0               ;;TERMINATOR
040102  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
040112  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
040117     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
040124  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
040132  070075     000
040135     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
040142  036440  000040
                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                ;;*************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

040146  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
040150  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
040156  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
040160  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
040162  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
040164  104407                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
040166  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
040170  010037  040274                  MOV     R0,5$           ;;AND SAVE IT
040174  005001                          CLR     R1              ;;CLEAR DATA WORD
040176  005002                          CLR     R2
040200  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
040202  001420                          BEQ     3$              ;;IF ZERO GET OUT
040204  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
040210  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
040212  122716  000067                  CMPB    #'7,(SP)
040216  002423                          BLT     4$
040220  006301                          ASL     R1              ;;*2
```

```
040222  006102                          ROL     R2
040224  006301                          ASL     R1              ;;*4
040226  006102                          ROL     R2
040230  006301                          ASL     R1              ;;*8
040232  006102                          ROL     R2
040234  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
040240  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
040242  000756                          BR      2$              ;;LOOP
040244  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
040246  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
040252  010237  040304                  MOV     R2,$HIOCT
040256  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
040260  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
040262  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
040264  000002                          RTI                     ;;RETURN
040266  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK

040270  105010                          CLRB    (R0)            ;;SET A TERMINATOR
040272  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR.
040274  000000                  5$:     .WORD   0
040276  104401  001206                  TYPE    ,$QUES          ;;"?" "CR" & "LF"
040302  000730                          BR      1$              ;;TRY AGAIN
040304  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
                                .SBTTL  TRAP DECODER

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

040306  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
040310  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
040314  005740                          TST     -(R0)           ;;BACKUP BY 2
040316  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
040320  006300                          ASL     R0              ;;POSITION FOR INDEXING
040322  016000  040342                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
040326  000200                          RTS     R0              ;;GO TO ROUTINE


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

040330  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
040332  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
040340  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
040342  040330                  $TRPAD: .WORD   $TRAP2
040344  036550                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
040346  037324                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
040350  037300                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
```

```
040352  037340                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
040354  036324                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)


040356  037526                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
040360  037646                          $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
040362  040146                          $RDOCT  ;;CALL=RDOCT    TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*****************************************************************
                                ;POWER DOWN ROUTINE
040364  012737  040524  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
040372  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
040400  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
040402  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
040404  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
040406  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
040410  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
040412  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
040414  017746  140514                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
040420  010637  040530                  MOV     SP,$SAVR6       ;;SAVE SP
040424  012737  040436  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
040432  000000                          HALT
040434  000776                          BR      .-2             ;;HANG UP

                                ;;*****************************************************************
                                ;POWER UP ROUTINE
040436  012737  040524  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
040444  013706  040530                  MOV     $SAVR6,SP       ;;GET SP
040450  005037  040530                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
040454  005237  040530          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
040460  001375                          BNE     1$              ;;OF  WORD
040462  012677  140446                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
040466  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
040470  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
040472  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
040474  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
040476  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
040500  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
040502  012737  040364  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
040510  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
040516  104401                          TYPE                    ;REPORT THE POWER FAILURE
040520  040532                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
040522  000002                          RTI
040524  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
040526  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
040530  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
040532  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
040540  000122
                                        .EVEN
                                ;;*****************************************************************
                                ;;*****************************************************************
                                ;;*****************************************************************

040542  005015  005015  040515  MSG1:   .ASCIZ<15><12><15><12>*MAINDEC-11-DQKAA-J  11/6X CACHE DIAGNOSTIC*<15><12><15><1
040550  047111  042504  026503
```

```
040556  030461  042055  045521
040564  040501  030455  020040
040572  030461  033057  020130
040600  040503  044103  020105
040606  044504  043501  047516
040614  052123  041511  005015
040622  005015     000
040625     015  050012  053517  MSG2:   .ASCIZ  <15><12>*POWER MACHINE DOWN AND THEN UP*<15><12>
040632  051105  046440  041501
040640  044510  042516  042040
040646  053517  020116  047101
040654  020104  044124  047105
040662  052440  006520  000012
040670  005015  054524  042520  MSG3:   .ASCII<CR><LF>*TYPE WHICH DEVICE SHOULD BE USED*<CR><LF>
040676  053440  044510  044103
040704  042040  053105  041511
040712  020105  044123  052517
040720  042114  041040  020105
040726  051525  042105  005015
040734  030012  055440  040503          .ASCII<LF>*0 [CARRIAGE RETURN]-UNIBUS EXERCISOR (M7855)*<CR><LF>
040742  051122  040511  042507
040750  051040  052105  051125
040756  056516  052455  044516
040764  052502  020123  054105
040772  051105  044503  047523
041000  020122  046450  034067
041006  032465  006451     012
041013     061  055440  040503          .ASCII*1 [CARRIAGE RETURN]-BUS TESTER (OLD)*<CR><LF>
041020  051122  040511  042507
041026  051040  052105  051125
041034  056516  041055  051525
041042  052040  051505  042524
041050  020122  047450  042114
041056  006451     012
041061     062  055440  040503          .ASCII*2 [CARRIAGE RETURN]-RK05*<CR><LF>
041066  051122  040511  042507
041074  051040  052105  051125
041102  056516  051055  030113
041110  006465     012
041113     063  055440  040503          .ASCII*3 [CARRIAGE RETURN]-RP03*<CR><LF>
041120  051122  040511  042507
041126  051040  052105  051125
041134  056516  051055  030120
041142  006463     012
041145     064  055440  040503          .ASCIZ*4 [CARRIAGE RETURN]-TU10*<CR><LF><CR><LF>
041152  051122  040511  042507
041160  051040  052105  051125
041166  056516  052055  030525
041174  006460  006412  000012
041202  005015  020077  044440  MSG4:   .ASCIZ<CR><LF>*?  INVALID ENTRY, TRY AGAIN*<CR><LF>
041210  053116  046101  042111
041216  042440  052116  054522
041224  020054  051124  020131
041232  043501  044501  006516
041240  000012
041242  005015  052040  050131  MSG5:   .ASCIZ<CR><LF>* TYPE THE UBE'S DATA BUFFER ADDRESS*<CR><LF>
```

```
 6385  041250  020105  044124  020105
 6386  041256  041125  023505  020123
 6387  041264  040504  040524  041040
 6388  041272  043125  042506  020122
 6389  041300  042101  051104  051505
 6390  041306  006523  000012
 6391  041312  005015  042040  053105  MSG6:   .ASCII<CR><LF>* DEVICE DOES NOT RESPOND*<CR><LF>
 6392  041320  041511  020105  047504
 6393  041326  051505  047040  052117
 6394  041334  051040  051505  047520
 6395  041342  042116  005015
 6396  041346  020040  020040  020040          .ASCIZ*         REFERENCE TO IT TRAPS TO 4*<CR><LF>
 6397  041354  020040  051040  043105
 6398  041362  051105  047105  042503
 6399  041370  052040  020117  052111
 6400  041376  052040  040522  051520
 6401  041404  052040  020117  006464
 6402  041412  000012
 6403  041414  005015  044127  041511  MSG7:   .ASCII<CR><LF>*WHICH DRIVE SHOULD BE USED?*<CR><LF>
 6404  041422  020110  051104  053111
 6405  041430  020105  044123  052517
 6406  041436  042114  041040  020105
 6407  041444  051525  042105  006477
 6408  041452     012
 6409  041453     124  050131  020105          .ASCIZ*TYPE 0-7<CARRIAGE RETURN>*<CR><LF>
 6410  041460  026460  036067  040503
 6411  041466  051122  040511  042507
 6412  041474  051040  052105  051125
 6413  041502  037116  005015     000
 6414  041507     015  052412  044516  MSG10:  .ASCIZ<CR><LF>*UNIT NOT SELECTED PROPERLY*<CR><LF>
 6415  041514  020124  047516  020124
 6416  041522  042523  042514  052103
 6417  041530  042105  050040  047522
 6418  041536  042520  046122  006531
 6419  041544  000012
 6420  041546  005015  047125  052111  MSG11:  .ASCIZ<CR><LF>*UNIT WRITE LOCK ON, SHOULD BE OFF*<CR><LF>
 6421  041554  053440  044522  042524
 6422  041562  046040  041517  020113
 6423  041570  047117  020054  044123
 6424  041576  052517  042114  041040
 6425  041604  020105  043117  006506
 6426  041612  000012
 6427  041614  005015  042504  044526  MSG12:  .ASCIZ<CR><LF>*DEVICE ERROR BIT SET*<CR><LF>
 6428  041622  042503  042440  051122
 6429  041630  051117  041040  052111
 6430  041636  051440  052105  005015
 6431  041644     000
 6432  041645     015  042012  053105  MSG13:  .ASCIZ<CR><LF>*DEVICE RDY BIT DOES NOT SET*<CR><LF>
 6433  041652  041511  020105  042122
 6434  041660  020131  044502  020124
 6435  041666  047504  051505  047040
 6436  041674  052117  051440  052105
 6437  041702  005015     000
 6438  041705     015  043012  051125  MSG14:  .ASCIZ<CR><LF>*FURTHER NPR DEVICE TESTS ABORTED*<CR><LF>
 6439  041712  044124  051105  047040
 6440  041720  051120  042040  053105
```

```
 6441  041726  041511  020105  042524
 6442  041734  052123  020123  041101
 6443  041742  051117  042524  006504
 6444  041750  000012
 6445
 6446  041752  051105  047522  035122  EM1:    .ASCIZ*ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE*
 6447  041760  052440  042516  050130
 6448  041766  041505  042524  020104
 6449  041774  040520  044522  054524
 6450  042002  042440  051122  051117
 6451  042010  044440  020116  040502
 6452  042016  045503  047111  020107
 6453  042024  052123  051117  000105
 6454  042032  051105  047522  035122  EM2:    .ASCIZ*ERROR: UNEXPECTED PARITY ERROR IN CACHE TAG*
 6455  042040  052440  042516  050130
 6456  042046  041505  042524  020104
 6457  042054  040520  044522  054524
 6458  042062  042440  051122  051117
 6459  042070  044440  020116  040503
 6460  042076  044103  020105  040524
 6461  042104  000107
 6462  042106  051105  047522  035122  EM3:    .ASCIZ*ERROR: UNEXPECTED PARITY ERROR IN CACHE DATA LOW*
 6463  042114  052440  042516  050130
 6464  042122  041505  042524  020104
 6465  042130  040520  044522  054524
 6466  042136  042440  051122  051117
 6467  042144  044440  020116  040503
 6468  042152  044103  020105  040504
 6469  042160  040524  046040  053517
 6470  042166     000
 6471  042167     105  051122  051117  EM4:    .ASCIZ*ERROR: UNEXPECTED PARITY ERROR IN CACHE DATA HIGH*
 6472  042174  020072  047125  054105
 6473  042202  042520  052103  042105
 6474  042210  050040  051101  052111
 6475  042216  020131  051105  047522
 6476  042224  020122  047111  041440
 6477  042232  041501  042510  042040
 6478  042240  052101  020101  044510
 6479  042246  044107     000
 6480  042251     106  052101  046101  EM5:    .ASCIZ*FATAL ERROR: CACHE CONTROL REG HELD WRONG DATA*
 6481  042256  042440  051122  051117
 6482  042264  020072  040503  044103
 6483  042272  020105  047503  052116
 6484  042300  047522  020114  042522
 6485  042306  020107  042510  042114
 6486  042314  053440  047522  043516
 6487  042322  042040  052101  000101
 6488  042330  040506  040524  020114  EM6:    .ASCIZ*FATAL ERROR: HIT/MISS REG HELD WRONG DATA*
 6489  042336  051105  047522  035122
 6490  042344  044040  052111  046457
 6491  042352  051511  020123  042522
 6492  042360  020107  042510  042114
 6493  042366  053440  047522  043516
 6494  042374  042040  052101  000101
 6495  042402  051105  047522  035122  EM7:    .ASCIZ*ERROR: DATA CACHED ON DATOB TO NO "HIT" ADDR.*
 6496  042410  042040  052101  020101
 042416  040503  044103  042105
 042424  047440  020116  040504
 042432  047524  020102  047524
 042440  047040  020117  044047
 042446  052111  020047  042101
 042454  051104  000056
 042460  051105  047522  035122  EM10:   .ASCIZ*ERROR: DATA NOT CACHED ON DATOB TO A 'HIT' ADDR.*
 042466  042040  052101  020101
 042474  047516  020124  040503
 042502  044103  042105  047440
 042510  020116  040504  047524
 042516  020102  047524  040440
 042524  023440  044510  023524
 042532  040440  042104  027122
 042540     000
 042541     105  051122  051117  EM11:   .ASCIZ*ERROR: CACHE DID NOT CONTAIN PROPER DATA ON DATOB*
 042546  070072  040503  044103
 042554  020105  044504  020104
 042562  047516  020124  047503
 042570  052116  044501  020116
 042576  051120  050117  051105
 042604  042040  052101  020101
 042612  047117  042040  052101
 042620  041117     000
 042623     105  051122  051117  EM12:   .ASCIZ*ERROR: FORCE MISS BIT FAILED TO CAUSE MISS*
 042630  020072  047506  041522
 042636  020105  044515  051523
 042644  041040  052111  043040
 042652  044501  042514  020104
 042660  047524  041440  052501
 042666  042523  046440  051511
 042674  000123
 042676  051105  047522  035122  EM14:   .ASCIZ*ERROR: ADDRESS COULD NOT BE MADE A 'HIT' AFTER DATO TO IT*
 042704  040440  042104  042522
 042712  051523  041440  052517
 042720  042114  047040  052117
 042726  041040  020105  040515
 042734  042504  040440  023440
 042742  044510  023524  040440
 042750  052106  051105  042040
 042756  052101  020117  047524
 042764  044440  000124
 042770  051105  047522  035122  EM16:   .ASCIZ*ERROR: UNEXPECTED TRAP TO VECTOR 4*
 042776  052440  042516  050130
 043004  041505  042524  020104
 043012  051124  050101  052040
 043020  020117  042526  052103
 043026  051117  032040     000
 043033     105  051122  051117  EM17:   .ASCIZ*ERROR: FORCE MISS DID NOT PREVENT CACHE TRACKING*
 043040  020072  047506  041522
 043046  020105  044515  051523
 043054  042040  042111  047040
 043062  052117  050040  042522
 043070  042526  052116  041440
 043076  041501  042510  052040
 043104  040522  045503  047111
```

```
 043112  000107
 043114  051105  047522  035122  EM20:   .ASCIZ*ERROR: PHYSICAL ADDRESS LINES ERROR*<15><12>*        ADDRESS HELD WRONG D
 043122  050040  054510  044523
 043130  040503  020114  042101
 043136  051104  051505  020123
 043144  044514  042516  020123
 043152  051105  047522  006522
 043160  020012  020040  020040
 043166  020040  040440  042104
 043174  042522  051523  044040
 043202  046105  020104  051127
 043210  047117  020107  040504
 043216  040524     000
 043221     105  051122  051117  EM21:   .ASCIZ*ERROR: TRAP TO VECTOR 4  WHEN TESTING PHYSICAL ADDRESS LINES*
 043226  020072  051124  050101
 043234  052040  020117  042526
 043242  052103  051117  032040
 043250  020040  044127  047105
 043256  052040  051505  044524
 043264  043516  050040  054510
 043272  044523  040503  020114
 043300  042101  051104  051505
 043306  020123  044514  042516
 043314  000123
 043316  051105  047522  035122  EM22:   .ASCIZ*ERROR:TEST OF ADDRESS COMPARATOR FAILED TO BE A MISS WHEN*
 043324  042524  052123  047440
 043332  020106  042101  051104
 043340  051505  020123  047503
 043346  050115  051101  052101
 043354  051117  043040  044501
 043362  042514  020104  047524
 043370  041040  020105  020101
 043376  044515  051523  053440
 043404  042510  000116
 043410  051105  047522  035122  EM23:   .ASCIZ*ERROR:TEST OF ADDRESS COMPARATOR FAILED TO BE A HIT WHEN*
 043416  042524  052123  047440
 043424  020106  042101  051104
 043432  051505  020123  047503
 043440  050115  051101  052101
 043446  051117  043040  044501
 043454  042514  020104  047524
 043462  041040  020105  020101
 043470  044510  020124  044127
 043476  047105     000
 043501     105  051122  051117  EM24:   .ASCIZ*ERROR:FORCE MISS DID NOT INHIBIT PARITY ERRORS*
 043506  043072  051117  042503
 043514  046440  051511  020123
 043522  044504  020104  047516
 043530  020124  047111  044510
 043536  044502  020124  040520
 043544  044522  054524  042440
 043552  051122  051117  000123
 043560  051105  047522  035122  EM25:   .ASCIZ*ERROR:DATO TO I/O ADDRESS WRITTEN IN CACHE*
 043566  040504  047524  052040
 043574  020117  027511  020117
 043602  042101  051104  051505
 043610  020123  051127  052111
 043616  042524  020116  047111
 043624  041440  041501  042510
 043632     000
 043633     105  051122  051117  EM26:   .ASCIZ*ERROR:CACHE CONTROL REG HELD WRONG DATA*
 043640  041472  041501  042510
 043646  041440  047117  051124
 043654  046117  051040  043505
 043662  044040  046105  020104
 043670  051127  047117  020107
 043676  040504  040524     000
 043703     105  051122  051117  EM27:   .ASCII*ERROR:TEST OF TAG PARITY GENERATOR/CHECKER FAILED*<15><12>
 043710  052072  051505  020124
 043716  043117  052040  043501
 043724  050040  051101  052111
 043732  020131  042507  042516
 043740  040522  047524  027522
 043746  044103  041505  042513
 043754  020122  040506  046111
 043762  042105  005015
 043766  020040  020040  020040          .ASCIZ*      DID NOT GET PARITY TRAP FROM TAG FIELD WHEN WROTE WRONG PARITY*
 043774  044504  020104  047516
 044002  020124  042507  020124
 044010  040520  044522  054524
 044016  052040  040522  020120
 044024  051106  046517  052040
 044032  043501  043040  042511
 044040  042114  053440  042510
 044046  020116  051127  052117
 044054  020105  051127  047117
 044062  020107  040520  044527
 044070  054524     000
 044073     105  051122  051117  EM31:   .ASCII*ERROR:TEST OF TAG PARITY GENERATOR/CHECKER FAILED*<15><12>
 044100  052072  051505  020124
 044106  043117  052040  043501
 044114  050040  051101  052111
 044122  020131  042507  042516
 044130  040522  047524  027522
 044136  044103  041505  042513
 044144  020122  040506  046111
 044152  042105  005015
 044156  020040  020040  020040          .ASCIZ*      TAG FIELD HELD WRONG DATA ON PARITY TRAP*
 044164  040524  020107  044506
 044172  046105  020104  042510
 044200  042114  053440  047522
 044206  043516  042040  052101
 044214  020101  047117  050040
 044222  051101  052111  020131
 044230  051124  050101     000
 044235     105  051122  051117  EM32:   .ASCII*ERROR:TEST OF TAG PARITY GENERATOR/CHECKER FAILED*<15><12>
 044242  052072  051505  020124
 044250  043117  052040  043501
 044256  050040  051101  052111
 044264  020131  042507  042516
 044272  040522  047524  027522
 044300  044103  041505  042513
```

```
 044306  020122  040506  046111
 044314  042105  005015
 044320  020040  020040  020040          .ASCIZ*      PARITY ERROR IN HIGH DATA BYTE*
 044326  040520  044522  054524
 044334  042440  051122  051117
 044342  044440  020116  044510
 044350  044107  042040  052101
 044356  020101  054502  042524
 044364     000
 044365     105  051122  051117  EM33:   .ASCII*ERROR:TEST OF TAG PARITY GENERATOR/CHECKER FAILED*<15><12>
 044372  052072  051505  020124
 044400  043117  052040  043501
 044406  050040  051101  052111
 044414  020131  042507  042516
 044422  040522  047524  027522
 044430  044103  041505  042513
 044436  020122  040506  046111
 044444  042105  005015
 044450  020040  020040  020040          .ASCIZ*      PARITY ERROR IN LOW DATA BYTE*
 044456  040520  044522  054524
 044464  042440  051122  051117
 044472  044440  020116  047514
 044500  020127  040504  040524
 044506  041040  052131  000105
 044514  051105  047522  035122  EM34:   .ASCII*ERROR:TEST OF TAG PARITY GENERATOR/CHECKER FAILED*<15><12>
 044522  042524  052123  047440
 044530  020106  040524  020107
 044536  040520  044522  054524
 044544  043440  047105  051105
 044552  052101  051117  041457
 044560  042510  045503  051105
 044566  043040  044501  042514
 044574  006504     012
 044577     040  020040  020040          .ASCIZ*      PARITY ERROR IN TAG FIELD*
 044604  050040  051101  052111
 044612  020131  051105  047522
 044620  020122  047111  052040
 044626  043501  043040  042511
 044634  042114     000
 044637     105  051122  051117  EM35:   .ASCII*ERROR:TEST OF DATA PARITY GENERATOR/CHECKER FAILED*<15><12>
 044644  052072  051505  020124
 044652  043117  042040  052101
 044660  020101  040520  044522
 044666  054524  043440  047105
 044674  051105  052101  051117
 044702  041457  042510  045503
 044710  051105  043040  044501
 044716  042514  006504     012
 044723     040  020040  020040          .ASCIZ*      NO PARITY TRAP WHEN WROTE WRONG PARITY*
 044730  047040  020117  040520
 044736  044522  054524  052040
 044744  040522  020120  044127
 044752  047105  053440  047522
 044760  042524  053440  047522
 044766  043516  050040  051101
 044774  052111  000131
```

```
045000  051105  047522  035122  EM36:   .ASCII*ERROR:TEST OF DATA PARITY GENERATOR/CHECKER FAILED*<15><12>
045006  042524  052123  047440
045014  020106  040504  040524
045022  050040  051101  052111
045030  020131  042507  042516
045036  040522  047524  027522
045044  044103  041505  042513
045052  020122  040506  046111
045060  042105  005015
045064  020040  020040  020040          .ASCIZ*      NO PARITY TRAP FROM LOW BYTE WHEN WROTE WRONG PARITY*
045072  047516  050040  051101
045100  052111  020131  051124
045106  050101  043040  047522
045114  020115  047514  020127
045122  054502  042524  053440
045130  042510  020116  051127
045136  052117  020105  051127
045144  047117  020107  040520
045152  044522  054524     000
045157     105  051122  051117  EM37:   .ASCII*ERROR:TEST OF DATA PARITY GENERATOR/CHECKER FAILED*<15><12>
045164  052072  051505  020124
045172  043117  042040  052101
045200  020101  040520  044522
045206  054524  043440  047105
045214  051105  052101  051117
045222  041457  042510  045503
045230  051105  043040  044501
045236  042514  006504     012
045243     040  020040  020040          .ASCIZ*      NO PARITY TRAP FROM HIGH BYTE WHEN WROTE WRONG PARITY*
045250  047040  020117  040520
045256  044522  054524  052040
045264  040522  020120  051106
045272  046517  044040  043511
045300  020110  054502  042524
045306  053440  042510  020116
045314  051127  052117  020105
045322  051127  047117  020107
045330  040520  044522  054524
045336     000
045337     105  051122  051117  EM40:   .ASCII*ERROR:TEST OF DATA PARITY GENERATOR/CHECKER FAILED*<15><12>
045344  052072  051505  020124
045352  043117  042040  052101
045360  020101  040520  044522
045366  054524  043440  047105
045374  051105  052101  051117
045402  041457  042510  045503
045410  051105  043040  044501
045416  042514  006504     012
045423     040  020040  020040          .ASCIZ*      PARITY ERROR IN LOW BYTE*
045430  050040  051101  052111
045436  020131  051105  047522
045444  020122  047111  046040
045452  053517  041040  052131
045460  000105
045462  051105  047522  035122  EM41:   .ASCII*ERROR:TEST OF DATA PARITY GENERATOR/CHECKER FAILED*<15><12>
045470  042524  052123  047440
```

```
045476  020106  040504  040524
045504  050040  051101  052111
045512  020131  042507  042516
045520  040522  047524  027522
045526  044103  041505  042513
045534  020122  040506  046111
045542  042105  005015
045546  020040  020040  020040          .ASCIZ*      PARITY ERROR IN HIGH BYTE*
045554  040520  044522  054524
045562  042440  051122  051117
045570  044440  020116  044510
045576  044107  041040  052131
045604  000105
045606  051105  047522  035122  EM42:   .ASCIZ*ERROR:NO PARITY TRAP FROM LOC WRITTEN WITH WRONG PARITY*
045614  047516  050040  051101
045622  052111  020131  051124

045630  050101  043040  047522
045636  020115  047514  020103
045644  051127  052111  042524
045652  020116  044527  044124
045660  053440  047522  043516
045666  050040  051101  052111
045674  000131
045676  051105  047522  035122  EM43:   .ASCIZ*ERROR: ADDRESS COULD NOT BE MADE A HIT*
045704  040440  042104  042522
045712  051523  041440  052517
045720  042114  047040  052117
045726  041040  020105  040515
045734  042504  010440  044040
045742  052111     000
045745     105  051122  051117  EM44:   .ASCIZ*ERROR: ADDRESS NOT INVALIDATED BY PARITY TRAP*
045752  020072  042101  051104
045760  051505  020123  047516
045766  020124  047111  040526
045774  044514  040504  042524
046002  020104  054502  050040
046010  051101  052111  020131
046016  051124  050101     000
046023     105  051122  051117  EM45:   .ASCIZ*ERROR: TAG PARITY ERROR WHEN TESTING TAG P BIT*
046030  020072  040524  020107
046036  040520  044522  054524
046044  042440  051122  051117
046052  053440  042510  020116
046060  042524  052123  047111
046066  020107  040524  020107
046074  020120  044502  000124
046102  051105  047522  035122  EM46:   .ASCIZ*ERROR: LOW BYTE PARITY ERROR WHEN TESTING TAG PARITY BIT*
046110  046040  053517  041040
046116  052131  020105  040520
046124  044522  054524  042440
046132  051122  051117  053440
046140  042510  020116  042524
046146  052123  047111  020107
046154  040521  020107  040520
046162  044522  054524  041040
046170  052111     000
```

```
046173     105  051122  051117  EM47:   .ASCIZ*ERROR: HIGH BYTE PARITY ERROR WHEN TESTING TAG P BIT*
046200  070072  044510  044107
046206  041040  052131  020105
046214  040520  044522  054524
046222  042440  051122  051117
046230  053440  042510  020116
046236  042524  052123  047111
046244  070107  040524  020107
046252  020120  044502  000124
046260  051105  047522  035122  EM50:   .ASCIZ*ERROR: HIGH BYTE PARITY ERROR WHEN TESTING DATA P BIT*
046266  044040  043511  020110
046274  054502  042524  050040
046302  051101  052111  020131
046310  051105  047522  020122
046316  044127  047105  052040
046324  051505  044524  043516
046332  042040  052101  020101
046340  070120  044502  000124
046346  051105  047522  035122  EM51:   .ASCIZ*ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA P BIT*
046354  046040  053517  041040
046362  052131  020105  040520
046370  044522  054524  042440
046376  051122  051117  053440
046404  042510  020116  042524
046412  052123  047111  020107
046420  040504  040524  050040
046426  041040  052111     000
046433     105  051122  051117  EM52:   .ASCIZ*ERROR: TAG PARITY ERROR WHEN TESTING TAG ADDRESS BITS*
046440  020072  040524  020107
046446  040520  044522  054524
046454  042440  051122  051117
046462  053440  042510  020116
046470  042524  052123  047111
046476  020107  040524  020107
046504  042101  051104  051505
046512  020123  044502  051524
046520     000
046521     105  051122  051117  EM53:   .ASCIZ*ERROR: LOW BYTE PARITY ERROR WHEN TESTING TAG ADDRESS BITS*
046526  020072  047514  020127
046534  054502  042524  050040
046542  051101  052111  020131
046550  051105  047522  020122
046556  044127  047105  052040
046564  051505  044524  043516
046572  052040  043501  040440
046600  042104  042522  051523
046606  041040  052111  000123
046614  051105  047522  035122  EM54:   .ASCIZ*ERROR: HIGH BYTE PARITY ERROR WHEN TESTING TAG ADDRESS BITS*
046622  044040  043511  020110
046630  054502  042524  050040
046636  051101  052111  020131
046644  051105  047522  020122
046652  044127  047105  052040
046660  051505  044524  043516
046666  052040  043501  040440
046674  042104  042522  051523
```

```
046702  041040  052111  000123
046710  051105  047522  035122  EM55:   .ASCII*ERROR: TEST OF TAG ADDRESS BITS FAILED*<15><12>
046716  052040  051505  020124
046724  043117  052040  043501
046732  040440  042104  042522
046740  051523  041040  052111
046746  020123  040506  046111
046754  042105  005015
046760  020040  020040  020040          .ASCIZ*       ADDRESS COULD NOT BE MADE A HIT*
046766  040440  042104  042522
046774  051523  041440  052517
047002  042114  047040  052117
047010  041040  020105  040515
047016  042504  040440  044040
047024  052111     000
047027     105  051122  051117  EM56:   .ASCIZ*ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA FIELD*
047034  020072  047514  020127
047042  054502  042524  050040
047050  051101  052111  020131
047056  051105  047522  020122
047064  044127  047105  052040
047072  051505  044524  043516
047100  042040  052101  020101
047106  044506  046105  000104
047114  051105  047522  035122  EM57:   .ASCIZ*ERROR: HIGH BYTE PARITY ERROR WHEN TESTING DATA FIELD*
047122  044040  043511  020110
047130  054502  042524  050040
047136  051101  052111  020131
047144  051105  047522  020122
047152  044127  047105  052040
047160  051505  044524  043516
047166  042040  052101  020101
047174  044506  046105  000104
047202  051105  047522  035122  EM60:   .ASCIZ*ERROR: TAG PARITY ERROR WHEN TESTING DATA FIELD*
047210  052040  043501  050040
047216  051101  052111  020131
047224  051105  047522  020122
047232  044127  047105  052040
047240  051505  044524  043516
047246  042040  052101  020101
047254  044506  046105  000104
047262  051105  047522  035122  EM61:   .ASCIZ*ERROR: CACHE DATA LOC HELD WRONG DATA*
047270  041440  041501  042510
047276  042040  052101  020101
047304  047514  020103  042510
047312  042114  053440  047522
047320  043516  042040  052101
047326  000101
047330  051105  047522  035122  EM62:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE DATA FIELD FAILED*<15><12>
047336  042524  052123  047440
047344  020106  051515  020102
047352  042101  051104  051505
047360  020123  040450  030061
047366  020051  047524  041440
047374  041501  042510  042040
047402  052101  020101  044506
```

```
047410  046105  020104  040506
047416  046111  042105  005015
047424  020040  020040  020040          .ASCIZ*     ADDRESS COULD NOT BE MADE HIT*
047432  042101  051104  051505
047440  020123  047503  046125
047446  020104  047516  020124
047454  042502  046440  042101
047462  020105  044510  000124
047470  051105  047522  035122  EM63:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE DATA FIELD FAILED*<15><12>
047476  042524  052123  047440
047504  020106  051515  020102
047512  042101  051104  051505
047520  020123  040450  030061
047526  020051  047524  041440
047534  041501  042510  042040
047542  052101  020101  044506
047550  046105  020104  040506
047556  046111  042105  005015
047564  020040  020040  020040          .ASCIZ*     ADDRESS HELD WRONG DATA*
047572  042101  051104  051505
047600  020123  042510  042114
047606  053440  047522  043516
047614  042040  052101  000101
047622  051105  047522  035122  EM64:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE DATA FIELD FAILED*<15><12>
047630  042524  052123  047440
047636  020106  051515  020102
047644  042101  051104  051505
047652  020123  040450  030061
047660  020051  047524  041440
047666  041501  042510  042040
047674  052101  020101  044506
047702  046105  020104  040506
047710  046111  042105  005015
047716  020040  020040  020040          .ASCIZ*     PARITY ERROR LOW BYTE*
047724  040520  044522  054524
047732  042440  051122  051117
047740  046040  053517  041040
047746  052131  000105
047752  051105  047522  035122  EM65:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE DATA FIELD FAILED*<15><12>
047760  042524  052123  047440
047766  020106  051515  020102
047774  042101  051104  051505
050002  020123  040450  030061
050010  020051  047524  041440
050016  041501  042510  042040
050024  052101  020101  044506
050032  046105  020104  040506
050040  046111  042105  005015
050046  020040  020040  020040          .ASCIZ*     PARITY ERROR HIGH BYTE*
050054  040520  044522  054524
050062  042440  051122  051117
050070  044040  043511  020110
050076  054502  042524     000
050103     105  051122  051117  EM66:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE DATA FIELD FAILED*<15><12>
050110  052072  051505  020124
050116  043117  046440  041123
```

```
050124  040440  042104  042522
050132  051523  024040  030501
050140  024460  052040  020117
050146  040503  044103  020105
050154  040504  040524  043040
050162  042511  042114  043040
050170  044501  042514  006504
050176     012
050177     040  020040  020040          .ASCIZ*     PARITY ERROR TAG*
050204  050040  051101  052111
050212  020131  051105  047522
050220  020122  040524  000107
050226  051105  047522  035122  EM67:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE ADDRESS FIELD FAILED*<15><12>
050234  042524  052123  047440
050242  020106  051515  020102
050250  042101  051104  051505
050256  020123  040450  030061
050264  020051  047524  041440
050272  041501  042510  040440
050300  042104  042522  051523
050306  043040  042511  042114
050314  043040  044501  042514
050322  006504     012
050325     040  020040  020040          .ASCIZ*     ADDRESS COULD NOT BE MADE A HIT*
050332  040440  042104  042522
050340  051523  041440  052517
050346  042114  047040  052117
050354  041040  020105  040515
050362  042504  040440  044040
050370  052111     000
050373     105  051122  051117  EM70:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE ADDRESS FIELD FAILED*<15><12>
050400  052072  051505  020124
050406  043117  046440  041123
050414  040440  042104  042522
050422  051523  024040  030501
050430  024460  052040  020117
050436  040503  044103  020105
050444  042101  051104  051505
050452  020123  044506  046105
050460  020104  040506  046111
050466  042105  005015
050472  020040  020040  020040          .ASCIZ*     TAG PARITY ERROR*
050500  040524  020107  040520
050506  044522  054524  042440
050514  051122  051117     000
050521     105  051122  051117  EM71:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE ADDRESS FIELD FAILED*<15><12>
050526  052072  051505  020124
050534  043117  046440  041123
050542  040440  042104  042522
050550  051523  024040  030501
050556  024460  052040  020117
050564  040503  044103  020105
050572  042101  051104  051505
050600  020123  044506  046105
050606  020104  040506  046111
050614  042105  005015
```

```
 050620  020040  020040  020040          .ASCIZ*      LOW BYTE PARITY ERROR*
 050626  047514  020127  054502
 050634  042524  050040  051101
 050642  052111  020131  051105
 050650  047522  000122
 050654  051105  047522  035122  EM72:   .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO CACHE ADDRESS FIELD FAILED*<15><12>
 050662  042524  052123  047440
 050670  020106  051515  020102
 050676  042101  051104  051505
 050704  020123  040450  030061
 050712  020051  047524  041440
 050720  041501  042510  040440
 050726  042104  042522  051523
 050734  043040  042511  042114
 050742  043040  044501  042514
 050750  006504     012
 050753     040  020040  020040          .ASCIZ*      HIGH BYTE PARITY ERROR*
 050760  044040  043511  020110
 050766  054502  042524  050040
 050774  051101  052111  020131
 051002  051105  047522  000122
 051010  051105  047522  035122  EM73:   .ASCII*ERROR:DYNAMIC TEST OF CACHE FAILED*<15><12>
 051016  054504  040516  044515
 051024  020103  042524  052123
 051032  047440  020106  040503
 051040  044103  020105  040506
 051046  046111  042105  005015
 051054  020040  020040  020040          .ASCIZ*      LOC HELD WRONG DATA*
 051062  047514  020103  042510
 051070  042114  053440  047522
 051076  043516  042040  052101
 051104  000101
 051106  051105  047522  035122  EM74:   .ASCII*ERROR:DYNAMIC TEST OF CACHE FAILED*<15><12>
 051114  054504  040516  044515
 051122  020103  042524  052123
 051130  047440  020106  040503
 051136  044103  020105  040506
 051144  046111  042105  005015
 051152  020040  020040  020040          .ASCIZ*      TRAP TO 10 OCCURRED*
 051160  051124  050101  052040
 051166  020117  030061  047440
 051174  041503  051125  042522
 051202  000104
 051204  051105  047522  035122  EM75:   .ASCII*ERROR:DYNAMIC TEST OF CACHE FAILED*<15><12>
 051212  054504  040516  044515
 051220  020103  042524  052123
 051226  047440  020106  040503
 051234  044103  020105  040506
 051242  046111  042105  005015
 051250  020040  020040  020040          .ASCIZ*      LOW BYTE PARITY ERROR*
 051256  047514  020127  054502
 051264  042524  050040  051101
 051272  052111  020131  051105
 051300  047522  000122
 051304  051105  047522  035122  EM76:   .ASCII*ERROR:DYNAMIC TEST OF CACHE FAILED*<15><12>
 051312  054504  040516  044515
```

```
 051320  020103  042524  052123
 051326  047440  020106  040503
 051334  044103  020105  040506
 051342  046111  042105  005015
 051350  020040  020040  020040          .ASCIZ*      HIGH BYTE PARITY ERROR*
 051356  044510  044107  041040
 051364  052131  020105  040520
 051372  044522  054524  042440
 051400  051122  051117     000
 051405     105  051122  051117  EM77:   .ASCII*ERROR:DYNAMIC TEST OF CACHE FAILED*<15><12>
 051412  042072  047131  046501
 051420  041511  052040  051505
 051426  020124  043117  041440
 051434  041501  042510  043040
 051442  044501  042514  006504
 051450     012
 051451     040  020040  020040          .ASCIZ*      TAG PARITY ERROR*
 051456  052040  043501  050040
 051464  051101  052111  020131
 051472  051105  047522  000122
 051500  051105  047522  035122  EM101:  .ASCIZ*ERROR:CACHE CONTROL REG NOT INITIALIZED BY POWER FAIL*
 051506  040503  044103  020105
 051514  047503  052116  047522
 051522  020114  042522  020107
 051530  047516  020124  047111
 051536  052111  040511  044514
 051544  042532  020104  054502
 051552  050040  053517  051105
 051560  043040  044501  000114
 051566  051105  047522  035122  EM102:  .ASCIZ*ERROR:POWER UP FAILED TO INVALIDATE CACHE*
 051574  047520  042527  020122
 051602  050125  043040  044501
 051610  042514  020101  047524
 051616  044440  053116  046101
 051624  042111  052101  020105
 051632  040503  044103  000105
 051640  051105  047522  035122  EM103:  .ASCIZ*ERROR:DEVICE ERROR BIT SET WHEN DOING NPR, DATO TO ADDRESS*
 051646  042504  044526  042503
 051654  042440  051122  051117
 051662  041040  052111  051440
 051670  052105  053440  042510
 051676  020116  047504  047111
 051704  020107  050116  026122
 051712  042040  052101  020117
 051720  047524  040440  042104
 051726  042522  051523     000
 051733     105  051122  051117  EM104:  .ASCIZ*ERROR:CACHE LOCATION NOT INVALIDATED BY NPR, DATO TO ADDRESS*
 051740  041472  041501  042510
 051746  046040  041517  052101
 051754  047511  020116  047516
 051762  020124  047111  040526
 051770  044514  040504  042524
 051776  020104  054502  047040
 052004  051120  020054  040504
 052012  047524  052040  020117
 052020  042101  051104  051505
```

```
052026  000123
052030  051105  047522  035122  EM105:  .ASCII*ERROR:DID NOT GET PARITY TRAP WHEN DID NPR. DATO TO ADDRESS*<CR><LF>
052036  044504  020104  047516
052044  020124  042507  020124
052052  040520  044522  054524
052060  052040  040522  020120
052066  044127  047105  042040
052074  042111  047040  051120
052102  020054  040504  047524
052110  052040  020117  042101
052116  051104  051505  006523
052124     012
052125     040  020040  020040          .ASCIZ*      WRITTEN WITH WRONG PARITY*
052132  053440  044522  052124
052140  047105  053440  052111
052146  020110  051127  047117
052154  020107  040520  044522
052162  054524     000
052165     105  051122  051117  EM107:  .ASCIZ*ERROR:CACHE DID NOT TRACK WHEN FORCE MISS ON*
052172  041472  041501  042510
052200  042040  042111  047040
052206  052117  052040  040522
052214  045503  053440  042510
052222  020116  047506  041522
052230  020105  044515  051523
052236  047440  000116
052242  051105  047522  035122  EM110:  .ASCIZ*ERROR:RETRY TO BACKING STORE NOT DONE ON CACHE PARITY TRAP*
052250  042522  051124  020131
052256  047524  041040  041501
052264  044513  043516  051440
052272  047524  042522  047040
052300  052117  042040  047117
052306  020105  047117  041440
052314  041501  042510  050040
052322  051101  052111  020131
052330  051124  050101     000
052335     105  051122  051117  EM111:  .ASCII*ERROR:TEST OF VALID BIT FAILED*<CR><LF>
052342  052072  051505  020124
052350  043117  053040  046101
052356  042111  041040  052111
052364  043040  044501  042514
052372  006504     012
052375     040  020040  020040          .ASCIZ*      LOC COULD NOT BE MADE A HIT*
052402  046040  041517  041440
052410  052517  042114  047040
052416  052117  041040  020105
052424  040515  042504  040440
052432  044040  052111     000
052437     105  051122  051117  EM112:  .ASCII*ERROR:TEST OF VALID BIT FAILED*<CR><LF>
052444  052072  051505  020124
052452  043117  053040  046101
052460  042111  041040  052111
052466  043040  044501  042514
052474  006504     012
052477     040  020040  020040          .ASCIZ*      LOC NOT INVALIDATED BY PARITY TRAP*
052504  046040  041517  047040
```

```
052512  052117  044440  053116
052520  046101  042111  052101
052526  042105  041040  020131
052534  040520  044522  054524
052542  052040  040522  000120
052550  051105  047522  035122  EM113:  .ASCIZ*ERROR:ADDRESS NOT INVALIDATED BY CONSOLE SWEEP*<CR><LF>
052556  042101  051104  051505
052564  020123  047516  020124
052572  047111  040526  044514
052600  040504  042524  020104
052606  054502  041440  047117
052614  047523  042514  051440
052622  042527  050105  005015
052630     000
052631     105  051122  051117  EM114:  .ASCIZ*ERROR:LOC WRITTEN WITH WRONG PARITY NOT INVALIDATED VIA NPR DATO*
052636  046072  041517  053440
052644  044522  052124  047105
052652  053440  052111  020110
052660  051127  047117  020107
052666  040520  044522  054524
052674  047040  052117  044440
052702  053116  046101  042111
052710  052101  042105  053040
052716  040511  047040  051120
052724  042040  052101  000117
052732  051105  047522  035122  EM115:  .ASCII*ERROR:PARITY TRAP WHILE TESTING LOC WRITTEN WITH WRONG PARITY*<CR><LF>
052740  040520  044522  054524
052746  052040  040522  020120
052754  044127  046111  020105
052762  042524  052123  047111
052770  020107  047514  020103
052776  051127  052111  042524
053004  020116  044527  044124
053012  053440  047522  043516
053020  050040  051101  052111
053026  006531     012
053031     040  020040  020040          .ASCIZ*      AND INVALIDATING IT VIA NPR DATO*
053036  040440  042116  044440
053044  053116  046101  042111
053052  052101  047111  020107
053060  052111  053040  040511
053066  047040  051120  042040
053074  052101  000117
053100  051105  047522  035122  EM116:  .ASCIZ*ERROR:CACHE ALLOCATED DURING ODD ADDRESS TRAP*
053106  040503  044103  020105
053114  046101  047514  040503
053122  042524  020104  052504
053130  044522  043516  047440
053136  042104  040440  042104
053144  042522  051523  052040
053152  040522  000120
053156  051105  047522  035122  EM117:  .ASCIZ*ERROR:CACHE ALLOCATED DURING RED ZONE TRAP*
053164  040503  044103  020105
053172  046101  047514  040503
053200  042524  020104  052504
053206  044522  043516  051040
```

```
 053214  042105  055040  047117
 053222  020105  051124  050101
 053230     000
 053231     105  051122  051117  EM120:  .ASCIZ*ERROR:CACHE ALLOCATED DURING KT ABORT*
 053236  041472  041501  042510
 053244  040440  046114  041517
 053252  052101  042105  042040
 053260  051125  047111  020107
 053266  052113  040440  047502
 053274  052122     000
 053277     105  051122  051117  EM121:  .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED*<CR><LF>
 053304  052072  051505  020124
 053312  043117  046440  041123
 053320  040440  042104  042522
 053326  051523  024040  030501
 053334  024460  052040  020117
 053342  040526  044514  020104
 053350  044502  020124  040506
 053356  046111  042105  005015
 053364  020040  020040  020040           .ASCIZ*       LOC NOT INVALIDATED*
 053372  047514  020103  047516
 053400  020124  047111  040526
 053406  044514  040504  042524
 053414  000104
 053416  051105  047522  035122  EM122:  .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED*<CR><LF>
 053424  042524  052123  047440
 053432  020106  051515  020102
 053440  042101  051104  051505
 053446  020123  040450  030061
 053454  020051  047524  053040
 053462  046101  042111  041040
 053470  052111  043040  044501
 053476  042514  006504     012
 053503     011  020040  020040           .ASCIZ*       PARITY ERROR TAG*
 053510  050040  051101  052111
 053516  020131  051105  047522
 053524  020122  040524  000107
 053532  051105  047522  035122  EM123:  .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED*<CR><LF>
 053540  042524  052123  047440
 053546  020106  051515  020102
 053554  042101  051104  051505
 053562  020123  040450  030061
 053570  020051  047524  053040
 053576  046101  042111  041040
 053604  052111  043040  044501
 053612  042514  006504     012
 053617     011  020040  020040           .ASCIZ*       PARITY ERROR LOW BYTE*
 053624  050040  051101  052111
 053632  020131  051105  047522
 053640  020122  047514  020127
 053646  054502  042524     000
 053653     105  051122  051117  EM124:  .ASCII*ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED*<CR><LF>
 053660  052072  051505  020124
 053666  043117  046440  041123
 053674  040440  042104  042522
 053702  051523  024040  030501
```

```
 053710  024460  052040  020117
 053716  040526  044514  020104
 053724  044502  020124  040506
 053732  046111  042105  005015
 053740  020011  020040  020040           .ASCIZ*       PARITY ERROR HIGH BYTE*
 053746  040520  044522  054524
 053754  042440  051122  051117
 053762  044040  043511  020110
 053770  054502  042524     000

 053775     120  036503  020040  DH1:    .ASCIZ*PC=   / P ADDH/ P ADDL/ PC OF PE*
 054002  027440  050040  040440
 054010  042104  027510  050040
 054016  040440  042104  027514
 054024  050040  020103  043117
 054032  050040  000105

 054036  041520  020075  020040  DH2:    .ASCIZ*PC=   / P ADDH/ P ADDL/ DATA/ PC OF PE*
 054044  020057  020120  042101
 054052  044104  020057  020120
 054060  042101  046104  020057
 054066  040504  040524  020057
 054074  041520  047440  020106
 054102  042520     000
 054105     120  036503  020040  DH5:    .ASCIZ*PC=   / DATA IS/DATA SHOULD BE*
 054112  027440  042040  052101
 054120  020101  051511  042057
 054126  052101  020101  044123
 054134  052517  042114  041040
 054142  000105
 054144  041520  020075  020040  DH6:    .ASCIZ*PC=   / DATA IS/DATA EXPECTED SET (0= DON'T CARE)*
 054152  020057  040504  040524
 054160  044440  027523  040504
 054166  040524  042440  050130
 054174  041505  042524  020104
 054202  042523  020124  030050
 054210  020075  047504  023516
 054216  020124  040503  042522
 054224  000051
 054226  041520  020075  020040  DH7:    .ASCIZ*PC=   /  P ADDH/ P ADDL*
 054234  020057  050040  040440
 054242  042104  027510  050040
 054250  040440  042104  000114
 054256  041520  020075  020040  DH11:   .ASCIZ*PC=   / P ADDH/ P ADDL/ DATA IS/ DATA SHOULD BE*
 054264  020057  020120  042101
 054272  044104  020057  020120
 054300  042101  046104  020057
 054306  040504  040524  044440
 054314  027523  042040  052101
 054322  020101  044123  052517
 054330  042114  041040  000105
 054336  041520  020075  020040  DH12:   .ASCIZ*PC=   / (CCR)  / P ADDH/ P ADDL*
 054344  020057  041450  051101
 054352  020051  027440  050040
 054360  040440  042104  027510
 054366  050040  040440  042104
 054374  000114
```

```
054376  041520  020075  020040  DH16:   .ASCIZ*PC=   /(CER)/PC WHEN TRAPPED*
054404  024057  042503  024522
054412  050057  020103  044127
054420  047105  052040  040522
054426  050120  042105     000
054433     120  036503  020040  DH21:   .ASCIZ*PC=   / P ADDH/ P ADDL/ (EREG)*
054440  027440  050040  040440
054446  042104  027510  050040
054454  040440  042104  027514
054462  024040  051105  043505
054470  000051
054472  041520  020075  020040  DH22:   .ASCIZ*PC=   / P ADDH/ P ADDL/ TAG FIELD=*
054500  020057  020120  042101
054506  044104  020057  020120
054514  042101  046104  020057
054522  040524  020107  044506
054530  046105  036504     000
054535     120  036503  020040  DH27:   .ASCIZ*PC=   / P ADDH/ P ADDL/ TAG SHOULD=*
054542  027440  050040  040440
054550  042104  027510  050040
054556  040440  042104  027514
054564  052040  043501  051440
054572  047510  046125  036504
054600     000
054601     120  036503  020040  DH30:   .ASCIZ*PC=   / P ADDH/ P ADDL/ (TAG)/ (TAG) SHOULD BE*
054606  027440  050040  040440
054614  042104  027510  050040
054622  040440  042104  027514
054630  024040  040524  024507
054636  020057  052050  043501
054644  020051  044123  052517
054652  042114  041040  000105
054660  041520  020075  020040  DH35:   .ASCIZ*PC=   / P ADDH/ P ADDL/ DATA SHOULD=*
054666  020057  020120  042101
054674  044104  020057  020120
054702  042101  046104  020057
054710  040504  040524  051440
054716  047510  046125  036504
054724     000
054725     120  036503  020040  DH45:   .ASCIZ*PC=   / P ADDH/ P ADDL/ DATA=*
054732  027440  050040  040440
054740  042104  027510  050040
054746  040440  042104  027514
054754  042040  052101  036501
054762     000
054763     120  036503  020040  DH100:  .ASCIZ*PC=   /DATA=*
054770  027440  040504  040524
054776  000075
055000  041520  000075          DH107:  .ASCIZ*PC=*
                                        .EVEN
055004  001116  001160  001162  DT1:    .WORD   $ERRPC,$REG1,$REG2,$REG3,$REG4,0
055012  001164  001166  000000
055020  001116  001160  001162  DT5:    .WORD   $ERRPC,$REG1,$REG2,0
055026  000000
055030  001116  001160  001162  DT12:   .WORD   $ERRPC,$REG1,$REG2,$REG3,0
055036  001164  000000
```

```
055042  001116  001160  000000  DT16:   .WORD   $ERRPC,$REG1,0
055050  001116  001160  001162  DT35:   .WORD   $ERRPC,$REG1,$REG2,$REG4,0
055056  001166  000000
055062  001116  001160  000000  DT100:  .WORD   $ERRPC,$REG1,0
055070  001116  000000          DT107:  .WORD   $ERRPC,0

                                ;;*************************************************************
                                ;;*************************************************************

                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;* THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB, THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.

                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

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

055074                          $ERRTB:

                                ;ITEM 1
055074  041752                          EM1                     ;ERROR: UNEXPECTED PARITY ERROR IN BACKING STORE
055076  053775                          DH1                     ;PC=/P ADDH /P ADDL /PC OF PE
055100  055030                          DT12                    ;$ERRPC,$REG1,$REG2,$REG3
055102  000000                          0
                                ;ITEM 2
055104  042032                          EM2                     ;ERROR:UNEXPECTED PARITY ERROR IN CACHE TAG
055106  054036                          DH2                     ;PC= /P ADDH /P ADDL /DATA /PC OF PE
055110  055004                          DT1                     ;$ERRPC,$REG1,$REG2,$REG3,$REG4
055112  000000                          0
                                ;ITEM 3
055114  042106                          EM3                     ;ERROR:UNEXPECTED PARITY ERROR IN CACHE DATA LOW
055116  054036                          DH2                     ;PC= /P ADDH /P ADDL /DATA /PC OF PE
055120  055004                          DT1                     ;$ERRPC,$REG1,$REG2,$REG3,$REG4
055122  000000                          0
                                ;ITEM 4
055124  042167                          EM4                     ;ERROR:UNEXPECTED PARITY ERROR IN CACHE DATA HIGH
055126  054036                          DH2                     ;PC= /P ADDH /P ADDL /DATA / PC OF PE
055130  055004                          DT1                     ;$ERRPC,$REG1,$REG2,$REG3,$REG4
055132  000000                          0
                                ;ITEM 5
055134  042251                          EM5                     ;FATAL ERROR: CACHE CONTROL REG HELD WRONG DATA
055136  054105                          DH5                     ;PC=/DATA IS /DATA SHOULD BE
055140  055020                          DT5                     ;$ERRPC,$REG1,$REG2
055142  000000                          0
                                ;ITEM 6
055144  042130                          EM6                     ;FATAL ERROR: HIT MISS REG HELD WRONG DATA
055146  054105                          DH5                     ;PC=/DATA IS/DATA SHOULD BE
055150  055020                          DT5                     ;$ERRPC,$REG1,$REG2
055152  000000                          0
                                ;ITEM 7
```

```
055154  042402                  EM7                     ;ERROR:DATA CACHED ON DATOB TO NO "HIT" ADDR..
055156  054226                  DH7                     ;PC=/P ADDH/P ADDL
055160  055020                  DT5                     ;$ERRPC,$REG1,$REG2
055162  000000                  0
                        ;ITEM 10
055164  042460                  EM10                    ;ERROR: DATA NOT CACHED ON DATOB TO A HIT LOC.
055166  054226                  DH7                     ;PC=/P ADDH/P ADDL
055170  055020                  DT5                     ;$ERRPC,$REG1,RREG2
055172  000000                  0
                        ;ITEM 11
055174  042541                  EM11                    ;ERROR:CACHE DID NOT CONTAIN PROPER DATA ON DATOB
055176  054256                  DH11                    ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
055200  055004                  DT1                     ;$ERRPC,$REG1,$REG2,$REG3,$REG4
055202  000000                  0
                        ;ITEM 12
055204  042623                  EM12                    ;ERROR: FORCE MISS BIT FAILED TO CAUSE MISS
055206  054336                  DH12                    ;PC=/(CCR)/P ADDH/P ADDL
055210  055030                  DT12                    ;$ERRPC,$REG1,$REG2,$REG3
055212  000000                  0
                        ;ITEM 13
055214  042330                  EM6                     ;FATAL ERROR: HIT MISS REG HELD WRONG DATA
055216  054144                  DH6                     ;PC=/DATA IS/DATA EXPECTED SET (0= DON'T CARE)
055220  055020                  DT5                     ;$ERRPC,$REG1,$REG2
055222  000000                  0
                        ;ITEM 14
055224  042676                  EM14                    ;ERROR: ADDRESS COULD NOT BE MADE A HIT AFTER DATO TO IT
055226  054226                  DH7                     ;PC=/P ADDH/P ADDL
055230  055020                  DT5                     ;$ERRPC,$REG1,$REG2
055232  000000                  0
                        ;ITEM 15
055234  000000                  0
055236  000000                  0
055240  000000                  0
055242  000000                  0
                        ;ITEM 16
055244  042770                  EM16                    ;ERROR: UNEXPECTED TRAP TO VECTOR 4
055246  054376                  DH16                    ;PC= /(CER)/PC WHEN TRAPPED
055250  055020                  DT5                     ;$ERRPC,$REG1,$REG2
055252  000000                  0
                        ;ITEM 17
055254  043033                  EM17                    ;ERROR: FORCE MISS DID NOT PREVENT CACHE TRACKING
055256  054226                  DH7                     ;PC=/P ADDH/P ADDL
055260  055020                  DT5                     ;$ERRPC,$REG1,$REG2
055262  000000                  0
                        ;ITEM 20
055264  043114                  EM20                    ;ERROR: PHYSICAL ADDRESS LINES ERROR
                                                        ;        ADDR. HELD WRONG DATA
055266  054256                  DH11                    ;PC=/P ADDH/P ADDL/DATA IS/ DATA SHOULD BE
055270  055004                  DT1                     ;$ERRPC,$REG1,$REG2,$REG3
055272  000000                  0
                        ;ITEM 21
055274  043721                  EM21                    ;ERROR: TRAP TO VECTOR 4 WHEN TESTING P.A. LINES
055276  054433                  DH21                    ;PC=/P ADDH/P ADDL/(EREG)
055300  055030                  DT12                    ;$ERRPC,$REG1,$REG2,$REG3
055302  000000                  0
                        ;ITEM 22
```

```
055304  043316                  EM22                    ;ERROR: TEST OF ADDR. COMPARATOR FAILED TO BE A MISS
055306  054472                  DH22                    ;PC=/P ADDH/P ADDL/TAG FIELD=
055310  055030                  DT12                    ;$ERRPC,$REG1,$REG2,$REG3
055312  000000                  0
                        ;ITEM 23
055314  043410                  EM23                    ;ERROR: TEST OF ADDR. COMPARATOR FAILED TO BE A HIT
055316  054472                  DH22                    ;PC=/P ADDH/P ADDL/TAG FIELD=
055320  055030                  DT12                    ;$ERRPC,$REG1,$REG2,$REG3
055322  000000                  0
                        ;ITEM 24
055324  043501                  EM24                    ;ERROR: FORCE MISS DID NOT INHIBIT PARITY ERRORS
055326  000000                  0
055330  000000                  0
055332  000000                  0
                        ;ITEM 25
055334  043560                  EM25                    ;ERROR: DATO TO I/O ADDRESS WRITTEN IN CACHE
055336  054226                  DH7                     ;PC=/P ADDH/P ADDL
055340  055020                  DT5                     ;$ERRPC,$REG1,$REG2
055342  000000                  0
                        ;ITEM 26
055344  043633                  EM26                    ;ERROR: CACHE CONTROL REG HOLD WRONG DATA
055346  054105                  DH5                     ;PC=/DATA IS /DATA SHOULD BE
055350  055020                  DT5                     ;$ERRPC,$REG1,$REG2
055352  000000                  0
                        ;ITEM 27
055354  043703                  EM27                    ;ERROR: TEST OF TAG PARITY GEN/CKER FAILED
                                                        ;          NO TAG PARITY TRAP WHEN WWP
055356  054535                  DH27                    ;PC=/P ADDH/P ADDL/(TAG) SHOULD BE
055360  055030                  DT12                    ;$ERRPC,$REG1,$REG2,$REG3
055362  000000                  0
                        ;ITEM 30
055364  043703                  EM27                    ;ERROR: TEST OF TAG PARITY GEN/CKER FAILED
                                                                 ; NO TAG PARITY TRAP WHEN WWP
055366  054601                  DH30                    ;PC=/P ADDH/P ADDL/(TAG)/(TAG) SHOULD BE
055370  055004                  DT1                     ;$ERRPC,$REG1,$REG2,$REG3,$REG4
055372  000000                  0
                        ;ITEM 31
055374  044073                  EM31                    ;ERROR: TEST OF TAG PARITY GEN/CKER FAILED
                                                        ;        (TAG) BAD ON PTRAP
055376  054601                  DH30                    ;PC=/P ADDH/P ADDL/(TAG)/(TAG) SHOULD BE
055400  055004                  DT1                     ;$ERRPC,$REG1,$REG2,$REG3
055402  000000                  0
                        ;ITEM 32
055404  044235                  EM32                    ;ERROR: TEST OF TAG PARITY GEN/CKER FAILED
                                                        ;          PARITY ERROR HIGH DATA BYTE
055406  054256                  DH11                    ;PC=/P ADDH/P ADDL/DATA IS/ DATA SHOULD BE
055410  055004                  DT1                     ;$ERRPC,$REG1,$REG2,$REG3
055412  000000                  0
                        ;ITEM 33
055414  044365                  EM33                    ;ERROR: TEST OF TAG PARITY GEN/CKER FAILED
                                                        ;          PARITY ERROR LOW DATA BYTE
055416  054256                  DH11                    ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
055420  055004                  DT1                     ;$ERRPC,$REG1,$REG2,$REG3
055422  000000                  0
                        ;ITEM 34
055424  044514                  EM34                    ;ERROR: TEST OF TAG PARITY GEN/CKER FAILED
```

```
                                                                 ;          PARITY ERROR IN TAG
      055426  054601                          DH30               ;PC=/P ADDH/P ADDL/(TAG)/(TAG) SHOULD BE
      055430  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055432  000000                          0
                                       ;ITEM 35
      055434  044637                          EM35               ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                 ;          NO PARITY TRAP OCCURRED
      055436  054660                          DH35               ;PC=/P ADDH/P ADDL/DATA SHOULD=
      055440  055050                          DT35               ;$ERRPC,$REG1,$REG2,$REG4
      055442  000000                          0
                                       ;ITEM 36
      055444  045000                          EM36               ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                 ;          NO PARITY TRAP FROM LOW BYTE WHEN WWP
      055446  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055450  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3,$REG4
      055452  000000                          0
                                       ;ITEM 37
      055454  045157                          EM37               ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                 ;          NO PARITY TRAP FROM HIGH BYTE WHEN WWP
      055456  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055460  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055462  000000                          0
                                       ;ITEM 40
      055464  045337                          EM40               ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                 ;          PARITY ERROR LOW BYTE
      055466  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055470  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055472  000000                          0
                                       ;ITEM 41
      055474  045462                          EM41               ;ERROR: TEST OF DATA PARITY GEN/CKER FAILED
                                                                 ;          PARITY ERROR HIGH BYTE
      055476  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055500  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055502  000000                          0
                                       ;ITEM 42
      055504  045606                          EM42               ;ERROR: NO PARITY TRAP FROM LOC WRITTEN WITH WRONG PARIT
      055506  054226                          DH7                ;PC=/P ADDH/P ADDL
      055510  055020                          DT5                ;$ERRPC,$REG1,$REG2
      055512  000000                          0
                                       ;ITEM 43
      055514  045676                          EM43               ;ERROR:ADDRESS COULD NOT BE MADE A HIT
      055516  054226                          DH7                ;PC=/P ADDH/P ADDL
      055520  055020                          DT5                ;$ERRPC,$REG1,$REG2
      055522  000000                          0
                                       ;ITEM 44
      055524  045745                          EM44               ;ERROR: ADDRESS NOT INVALIDATED BY PARITY TRAP
      055526  054226                          DH7                ;PC=/P ADDH/P ADDL
      055530  055020                          DT5                ;$ERRPC,$REG1,$REG2
      055532  000000                          0
                                       ;ITEM 45
      055534  046023                          EM45               ;ERROR: TAG PARITY ERROR WHEN TESTING TAG P BIT
      055536  054725                          DH45               ;PC= /P ADDH/P ADDL/DATA IS
      055540  055030                          DT12               ;$ERRPC, $REG1, $REG2, $REG3
      055542  000000                          0
                                       ;ITEM 46
      055544  046102                          EM46               ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING TAG P BIT
```

```
      055546  054725                          DH45               ;PC=/P ADDH/P ADDL/DATA=
      055550  055030                          DT12               ;$ERRPC, $REG1, $REG2, $REG3
      055552  000000                          0
                                       ;ITEM 47
      055554  046173                          EM47               ;ERROR: HIGH BYTE PARITY ERROR WHEN TESTING TAG P BIT
      055556  054725                          DH45               ;PC=/P ADDH/P ADDL/DATA=
      055560  055030                          DT12               ;$ERRPC, $REG1, $REG2, $REG3
      055562  000000                          0
                                       ;ITEM 50
      055564  046260                          EM50               ;ERROR: HIGH BYTE PARITY ERROR WHEN TESTING DATA P BIT
      055566  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055570  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055572  000000                          0
                                       ;ITEM 51
      055574  046346                          EM51               ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA P BIT
      055576  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055600  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055602  000000                          0
                                       ;ITEM 52
      055604  046433                          EM52               ;ERROR: TAG PARITY ERROR WHEN TESTING TAG ADDR. BITS
      055606  054601                          DH30               ;PC=/P ADDH/P ADDL/(TAG)/(TAG) SHOULD BE
      055610  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055612  000000                          0
                                       ;ITEM 53
      055614  046521                          EM53               ;ERROR: LOW BYTE PAR. ERROR WHEN TESTING TAG ADDR. BITS
      055616  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055620  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055622  000000                          0
                                       ;ITEM 54
      055624  046614                          EM54               ;ERROR: HIGH BYTE PAR. ERROR WHEN TESTING TAG ADDR. BITS
      055626  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055630  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055632  000000                          0
                                       ;ITEM 55
      055634  046710                          EM55               ;ERROR: TEST OF TAG ADDR. BITS FAILED
                                                                 ;          ADDR. COULD NOT BE MADE A HIT
      055636  054535                          DH27               ;PC=/P ADDH/P ADDL/(TAG) SHOULD=
      055640  055030                          DT12               ;$ERRPC, $REG1, $REG2, $REG3
      055642  000000                          0
                                       ;ITEM 56
      055644  047027                          EM56               ;ERROR: LOW BYTE PARITY ERROR WHEN TESTING DATA FIELD
      055646  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055650  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055652  000000                          0
                                       ;ITEM 57
      055654  047114                          EM57               ;ERROR: HIGH BYTE PARITY ERROR WHEN TESTING DATA FIELD
      055656  054256                          DH11               ;PC=/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
      055660  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055662  000000                          0
                                       ;ITEM 60
      055664  047202                          EM60               ;ERROR: TAG PARITY ERROR WHEN TESTING DATA FIELD
      055666  054601                          DH30               ;PC=/P ADDH/P ADDL/(TAG)/(TAG) SHOULD BE
      055670  055004                          DT1                ;$ERRPC,$REG1,$REG2,$REG3
      055672  000000                          0
                                       ;ITEM 61
      055674  047262                          EM61               ;ERROR: CACHE DATA LOC HELD WRONG DATA
```

```
 055676  054256                DH11            ;PC*/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
 055700  055004                DT1             ;$ERRPC,$REG1,$REG2,$REG3
 055702  000000                0

                       ;ITEM 62
 055704  047330                EM62            ;ERROR: TEST OF MSB ADDRESS (A10) TO  DATA FIELD FAILED
                                               ;       ADDRESS COULD NOT BE MADE A HIT
 055706  054226                DH7             ;PC*/P ADDH/P ADDL
 055710  055020                DT5             ;$ERRPC,$REG1,$REG2
 055712  000000                0

                       ;ITEM 63
 055714  047470                EM63            ;ERROR: TEST OF MSB ADDRESS (A10) TO  DATA FIELD FAILED
                                               ;       ADDRESS HELD WRONG DATA
 055716  054256                DH11            ;PC*/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
 055720  055004                DT1             ;$ERRPC,$REG1,$REG2,$REG3
 055722  000000                0

                       ;ITEM 64
 055724  047622                EM64            ;ERROR: TEST OF MSB ADDRESS (A10) TO  DATA FIELD FAILED
                                               ;       PARITY ERROR LOW BYTE
 055726  054725                DH45            ;PC*/P ADDH/P ADDL/DATA*
 055730  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 055732  000000                0

                       ;ITEM 65
 055734  047752                EM65            ;ERROR: TEST OF MSB ADDRESS (A10) TO  DATA FIELD FAILED
                                               ;       PARITY ERROR HIGH BYTE
 055736  054725                DH45            ;PC*/P ADDH/P ADDL/DATA*
 055740  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 055742  000000                0

                       ;ITEM 66
 055744  050103                EM66            ;ERROR: TEST OF MSB ADDRESS (A10) TO  DATA FIELD FAILED
                                               ;       PARITY ERROR TAG
 055746  054725                DH45            ;PC*/P ADDH/P ADDL/DATA*
 055750  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 055752  000000                0

                       ;ITEM 67
 055754  050226                EM67            ;ERROR: TEST OF MSB ADDRESS (A10) TO  TAG FIELD FAILED
                                               ;       ADDRESS COULD NOT BE MADE A HIT
 055756  054226                DH7             ;PC*/P ADDH/P ADDL
 055760  055020                DT5             ;$ERRPC,$REG1,$REG2
 055762  000000                0

                       ;ITEM 70
 055764  050373                EM70            ;ERROR: TEST OF MSB ADDRESS (A10) TO  TAG FIELD FAILED
                                               ;       TAG PARITY ERROR
 055766  054472                DH22            ;PC*/P ADDH/P ADDL/TAG FIELD*
 055770  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 055772  000000                0

                       ;ITEM 71
 055774  050521                EM71            ;ERROR: TEST OF MSB ADDRESS (A10) TO  TAG FIELD FAILED
                                               ;       LOW BYTE PARITY ERROR
 055776  054725                DH45            ;PC*/P ADDH/P ADDL/DATA*
 056000  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 056002  000000                0

                       ;ITEM 72
 056004  050654                EM72            ;ERROR: TEST OF MSB ADDRESS (A10) TO  TAG FIELD FAILED
                                               ;       HIGH BYTE PARITY ERROR
 056006  054725                DH45            ;PC*/P ADDH/P ADDL/DATA*
 056010  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
```

```
 056012  000000                0

                       ;ITEM 73
 056014  051010                EM73            ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                               ;       LOC HELD WRONG DATA
 056016  054256                DH11            ;PC*/P ADDH/P ADDL/DATA IS/DATA SHOULD BE
 056020  055004                DT1             ;$ERRPC,$REG1,$REG2,$REG3
 056022  000000                0

                       ;ITEM 74
 056024  051106                EM74            ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                               ;       TRAP TO 10 OCCURRED
 056026  053775                DH1             ;PC*/P ADDH/P ADDL/PC OF PE
 056030  055004                DT1             ;$ERRPC,$REG1,$REG2,$REG3
 056032  000000                0

                       ;ITEM 75
 056034  051204                EM75            ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                               ;       LOW BYTE PARITY ERROR
 056036  054725                DH45            ;PC*/P ADDH/P ADDL/DATA*
 056040  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 056042  000000                0

                       ;ITEM 76
 056044  051304                EM76            ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                               ;       HIGH BYTE PARITY ERROR
 056046  054725                DH45            ;PC*/P ADDH/P ADDL/DATA*
 056050  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 056052  000000                0

                       ;ITEM 77
 056054  051405                EM77            ;ERROR: DYNAMIC TEST OF CACHE FAILED
                                               ;       TAG PARITY ERROR
 056056  054472                DH22            ;PC*/P ADDH/P ADDL/TAG FIELD*
 056060  055030                DT12            ;$ERRPC, $REG1, $REG2, $REG3
 056062  000000                0

                       ;ITEM 100
 056064  000000                0
 056066  000000                0
 056070  000000                0
 056072  000000                0

                       ;ITEM 101
 056074  051500                EM101           ;ERROR: CACHE CONTROL REG NOT INITIALIZED BY POWER FAIL
 056076  054763                DH100           ;PC*/DATA*
 056100  055062                DT100           ;$ERRPC,$REG1
 056102  000000                0

                       ;ITEM 102
 056104  051566                EM102           ;ERROR: POWER UP FAILED TO INVALIDATE CACHE
 056106  055000                DH107           ;PC*
 056110  055070                DT107           ;$ERRPC
 056112  000000                0

                       ;ITEM 103
 056114  051640                EM103           ;ERROR: DEVICE ERROR BIT SET WHEN DOING NPR.DATO TO ADDR
 056116  054226                DH7             ;PC*/P ADDH/P ADDL
 056120  055020                DT5             ;$ERRPC,$REG1,$REG2
 056122  000000                0

                       ;ITEM 104
 056124  051733                EM104           ;ERROR: CACHE LOC NOT INVALIDATED BY NPR. DATO
 056126  054226                DH7             ;PC*/P ADDH/P ADDL
 056130  055020                DT5             ;$ERRPC,$REG1,$REG2
 056132  000000                0
```

```
                                ;ITEM 105
 056134  052030                         EM105                   ;ERROR: DID NOT GET PARITY TRAP WHEN DID NPR
                                                                ;       DATO TO ADDR, WRITTEN WITH WRONG PARITY
 056136  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056140  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056142  000000                         0
                                ;ITEM 106
 056144  000000                         0
 056146  000000                         0
 056150  000000                         0
 056152  000000                         0
                                ;ITEM 107
 056154  052165                         EM107                   ;ERROR: CACHE DID NOT TRACK WHEN FORCE MISS ON
 056156  055000                         DH107                   ;PC=
 056160  055070                         DT107                   ;$ERRPC
 056162  000000                         0
                                ;ITEM 110
 056164  052242                         EM110                   ;ERROR: RETRY TO BACKING STORE NOT DONE ON CACHE PARITY
 056166  055000                         DH107                   ;PC=
 056170  055070                         DT107                   ;$ERRPC
 056172  000000                         0
                                ;ITEM 111
 056174  052335                         EM111                   ;ERROR: TEST OF VALID BIT FAILED
                                                                ;       LOC COULD NOT BE MADE A HIT
 056176  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056200  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056202  000000                         0
                                ;ITEM 112
 056204  052437                         EM112                   ;ERROR: TEST OF VALID BIT FAILED
                                                                ;       LOC NOT INVALIDATED BY P TRAP
 056206  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056210  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056212  000000                         0
                                ;ITEM 113
 056214  052550                         EM113                   ;ERROR: ADDR. NOT INVALIDATED BY CONSOLE SWEEP
 056216  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056220  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056222  000000                         0
                                ;ITEM 114
 056224  052631                         EM114                   ;ERROR: LOC WRITTEN WITH WRONG PARITY NOT
                                                                ;       INVALIDATED VIA NPR DATO
 056226  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056230  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056232  000000                         0
                                ;ITEM 115
 056234  052732                         EM115                   ;ERROR: PARITY TRAP WHILE TESTING LOC
                                                                ;       WRITTEN WITH WRONG PARITY AND
                                                                ;       INVALIDATING VIA NPR DATO
 056236  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056240  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056242  000000                         0
                                ;ITEM 116
 056244  053100                         EM116                   ;ERROR: CACHE ALLOCATED DURING ODD ADDRESS TRAP
 056246  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056250  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056252  000000                         0
```

```
                                ;ITEM 117
 056254  053156                         EM117                   ;ERROR: CACHE ALLOCATED DURING RED ZONE TRAP
 056256  055000                         DH107                   ;PC=
 056260  055070                         DT107                   ;$ERRPC
 056262  000000                         0
                                ;ITEM 120
 056264  053231                         EM120                   ;ERROR: CACHE ALLOCATED DURING KT ABORT
 056266  055000                         DH107                   ;PC=
 056270  055070                         DT107                   ;$ERRPC
 056272  000000                         0
                                ;ITEM 121
 056274  053277                         EM121                   ;ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED
                                                                ;       LOC NOT INVALIDATED
 056276  054226                         DH7                     ;PC=/P ADDH/P ADDL
 056300  055020                         DT5                     ;$ERRPC,$REG1,$REG2
 056302  000000                         0

                                ;ITEM 122
 056304  053416                         EM122                   ;ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED
                                                                ;       PARITY ERROR TAG
 056306  054725                         DH45                    ;PC=/P ADDH/P ADDL/DATA=
 056310  055030                         DT12                    ;$ERRPC,$REG1,$REG2,$REG3
 056312  000000                         0
                                ;ITEM 123
 056314  053532                         EM123                   ;ERROR:TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED
                                                                ;       PARITY ERROR LOW BYTE
 056316  054725                         DH45                    ;PC=/P ADDH/P ADDL/DATA=
 056320  055030                         DT12                    ;$ERRPC,$REG1,$REG2,$REG3
 056322  000000                         0
                                ;ITEM 124
 056324  053653                         EM124                   ;ERROR: TEST OF MSB ADDRESS (A10) TO VALID BIT FAILED
                                                                ;       PARITY ERROR HIGH BYTE
 056326  054725                         DH45                    ;PC=/P ADDH/P ADDL/DATA=
 056330  055030                         DT12                    ;$ERRPC,$REG1,$REG2,$REG3
 056332  000000                         0
                                ;;*****************************************************************
                                ;TEST BUFFER
                                ;;*****************************************************************
         000001                         .END
```

```
MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 144
DQKKA.P11     07-FEB-77 11:01           CROSS REFERENCE TABLE -- USER SYMBOLS

ABASE = 000000           458      499
ACDW1 = 000000           458      501
ACDW2 = 000000           458      502
ACPUOP= 000000           458      473
ADDW0 = 000000           458      503
ADDW1 = 000000           458      504
ADDW10= 000000           458      513
ADDW11= 000000           458      514
ADDW12= 000000           458      515
ADDW13= 000000           458      516
ADDW14= 000000           458      517
ADDW15= 000000           458      518
ADDW2 = 000000           458      505
ADDW3 = 000000           458      506
ADDW4 = 000000           458      507
ADDW5 = 000000           458      508
ADDW6 = 000000           458      509
ADDW7 = 000000           458      510
ADDW8 = 000000           458      511
ADDW9 = 000000           458      512
ADD1H   031622          4918*    4921#    4997     5003
ADD1L   031620          4917*    4920#    4930     4939     4956*    4957*    4962*    4963*    4971*    4972*    4984     4990     5004
ADD2H   032444          5071*    5075#    5125     5131
ADD2L   032442          5072*    5074#    5084     5126     5132
ADEVCT= 000000           458      464
ADEVM = 000000           458      500
AFNV  = 000000           458      469
AENVM = 000000           458      470
AFATAL= 000000           458      461
AMADR1= 000000           458      486
AMADR2= 000000           458      490
AMADR3= 000000           458      493
AMADR4= 000000           458      496
AMAMS1= 000000           458      480
AMAMS2= 000000           458      488
AMAMS3= 000000           458      491
AMAMS4= 000000           458      494
AMSGAD= 000000           458      466
AMSGLG= 000000           458      467
AMSGTY= 000000           458      460
AMTYP1= 000000           458      481
AMTYP2= 000000           458      489
AMTYP3= 000000           458      492
AMTYP4= 000000           458      495
APASS = 000000           458      463
APRIOR= 000000           458
APTCSU= 000040          5901     6006#
APTENV= 000001          5641     5894     5962     6004#
APTSIZ= 000200           612     6003#
APTSPO= 000100          5896     5964     6005#
ASWREG= 000000           458      471
ATESTN= 000000           458      462
AUNIT = 000000           458      465
AUSWR = 000000           458      472
AVECT1= 000000           458      497
AVECT2= 000000           458      498


MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 145
DQKKA.P11     07-FEB-77 11:01           CROSS REFERENCE TABLE -- USER SYMBOLS

A1      003402           938      945#    1031     1039     1047
A10     003706          1013     1019#
A11     003752          1030     1033#
A12     003730          1020     1027#
A13     004002          1039     1041#
A14     004052          1034     1055#
A15     003214           900      902#
A16     003250           909      911#
A17     004054           910      985     1005     1059#
A3      003312           917      922#
A4      003334           918      927      932#
A5      003300           916#     920
A6      003360           935      937#
A77     003550           936      942      989#
A8      003346           934#     943
A80     003604           992      996#
A81     003606           994      997#
A82     003616           995      999#
A85     004042          1042     1053#
A86     004032          1046     1051#
BIT0  = 000001           124#
BIT00 = 000001           114#     124
BIT01 = 000002           113#     123
BIT02 = 000004           112#     122
BIT03 = 000010           111#     121
BIT04 = 000020           110#     120
BIT05 = 000040           109#     119
BIT06 = 000100           108#     118
BIT07 = 000200           107#     117
BIT08 = 000400           106#     116
BIT09 = 001000           105#     115     5584     5651
BIT1  = 000002           123#
BIT10 = 002000           104#    5629
BIT11 = 004000           103#    5591
BIT12 = 010000           102#
BIT13 = 020000           101#    5636
BIT14 = 040000           100#    5566
BIT15 = 100000            99#
BIT2  = 000004           122#
BIT3  = 000010           121#
BIT4  = 000020           120#
BIT5  = 000040           119#
BIT6  = 000100           118#
BIT7  = 000200           117#
BIT8  = 000400           116#
BIT9  = 001000           115#
BPTVEC= 000014           131#
BSD   = 055016           294#    3552     3722
BUFH  = 062000           296#    1124     1156     1211     1241     1243     1271*    1272     1277     1281*    1282*    1283     128
                        1292     1294     1297     1303*    1304*    1305     1307     1310     1331*    1332     1337     1834     184
                        1844     1888     1892*    1894     1942*    1943     2511     2520     2548     2557     2563     2584     259
                        2629     2702     2709     2710     2713     2714     2733     3158     3163     3653     5351
BUFL  = 060000           295#     296      844     1120     1380     1386     1526*    1528     1537     1560*    1562     2031     203
                        2045     2047     2072     2081     2087     2108     2121     2127     2129     2158     2232     2239     224
                        2243     2244     2263     2372     2377     3018     3098     3483     4219*    4220*    4232     4242     424
                        4325     4330*    4344     4356     4396     4409     4410     4420     4421     4598*    4600     4613     463
```

```
MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 146
DQKKAA.P11     07-FEB-77 11:01          CROSS REFERENCE TABLE -- USER SYMBOLS

                                4701    4803*   4917    4957    4963    4972    5347    5546    5548
B1       001734                  641     645#
B2       001710                  636#    643
CCR      = 177746                286#    354*    646*    896*   1103*   1105*   1107    1142*   1145*   1146    1149*   1162*   1188*
                                1195*   1207*   1210*   1213*   1217    1233*   1236*   1240*   1242*   1247*   1251*   1266*   1269*
                                1275*   1286*   1295*   1301*   1308*   1324*   1327*   1335*   1341*   1346*   1367*   1370*   1371
                                1374*   1384*   1390*   1394    1421*   1457*   1469*   1475*   1484*   1512*   1516*   1517    1520*
                                1524*   1527*   1532*   1544*   1559*   1561*   1567*   1576*   1618*   1633*   1636*   1641*   1659*
                                1728*   1736*   1745*   1830*   1839*   1841*   1843*   1845*   1851*   1891*   1893*   1896*   1901*
                                1941*   1949*   1981*   2026*   2030*   2054*   2056*   2088*   2090*   2092*   2116*   2135*   2139*
                                2141*   2143*   2164*   2172*   2178*   2185*   2191*   2227*   2233*   2261*   2269*   2326*   2330*
                                2364*   2370*   2393*   2409*   2412*   2418*   2507*   2510*   2524*   2530*   2532*   2564*   2566*
                                2568*   2592*   2606*   2610*   2612*   2614*   2635*   2642*   2648*   2655*   2661*   2697*   2703*
                                2731*   2739*   2796*   2800*   2807*   2844*   2875*   2907*   2975*   2987*   3012*   3019*   3040*
                                3081*   3107*   3117*   3150*   3156*   3179*   3193*   3249*   3289*   3320*   3352*   3420*   3432*
                                3441*   3475*   3481*   3523*   3591*   3598*   3603*   3611*   3645*   3651*   3693*   3761*   3768*
                                3773*   3799*   3806*   3808*   3833*   3840*   3847*   3853*   3859*   3947*   3977*   3992*   3996*
                                4004*   4080*   4096*   4114*   4122*   4126*   4131*   4139*   4214*   4217*   4240*   4246    4262*
                                4265*   4292*   4298*   4316*   4329*   4354*   4360*   4386*   4392*   4456*   4485*   4510*   4519*
                                4590*   4594*   4595    4597*   4599*   4616*   4619*   4626    4627*   4632*   4640*   4658*   4661*
                                4697*   4700*   4737*   4744*   4772*   4779*   4808    4822*   4831*   4906*   4912*   5039*   5048*
                                5192*   5251*
CDH      = 000106                287#   1796    1877    1927    2321    2457    2791    2970    3087    3232    3415    3573    3743
                                3909    4054    4559    5213
CDL      = 000106                288#   1785    1868    1916    2314    2465    2784    2963    3075    3240    3408    3565    3735
                                3901    4046    4170    4179    4550    5233
CER      = 177766                291#   5253
CR       = 000015                 39#    334#   5940    5950    6342    6348    6356    6363    6368    6373    6378    6384    6391
                                6396    6403    6409    6414    6420    6427    6432    6438    7178    7205    7217    7230    7250
                                7291    7305    7310    7332
CREG1    001212                  444#    690*    696     711*    743*    785*    834*   4914    5052    5373    5385*
CREG2    001214                  445#    695*    741*    746     781*    782*    836*    837*    841    4782*   4796*   5386*   5390*
                                5440    5490    5512    5520
CREG3    001216                  446#    692*   5384*
CREG4    001220                  447#    688*   5388*
CREG5    001222                  448#    686*   5387*
CREG6    001224                  449#    684*
CRLF     = 000200                 40#   5911    5950
CTAG     = 000107                289#   3095
DDISP    = 177570                 46#    418     600
DH1      053775                 7347#   7476    7795
DH100    054763                 7438#   7823
DH107    055000                 7441#   7828    7854    7859    7899    7904
DH11     054256                 7379#   7516    7552    7606    7612    7630    7636    7642    7648    7683    7688    7698    7703
                                7714    7719    7729    7741    7789
DH12     054336                 7387#   7521
DH16     054376                 7393#   7541
DH2      054036                 7353#   7481    7486    7491
DH21     054433                 7398#   7557
DH22     054472                 7404#   7562    7567    7771    7813
DH27     054535                 7410#   7588    7709
DH30     054601                 7417#   7594    7600    7618    7693    7724
DH35     054660                 7425#   7624
DH45     054725                 7432#   7660    7673    7678    7747    7753    7759    7777    7783    7801    7807    7916    7922
                                7928
DH5      054105                 7360#   7496    7501    7502


MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 147
DQKKAA.P11     07-FEB-77 11:01          CROSS REFERENCE TABLE -- USER SYMBOLS

DH6      054144                 7366#   7526
DH7      054226                 7375#   7506    7511    7531    7546    7577    7653    7658    7663    7735    7765    7833    7838
                                7844    7865    7871    7876    7882    7889    7894    7918
DISPLA   001136                  418#    600*    608*   5606*   5620*
DISPRE   000174                  345#    608
DSWP     = 177570                 45#    417     599
DT1      055004                 7443#   7482    7487    7492    7517    7553    7595    7601    7607    7613    7619    7631    7637
                                7643    7649    7684    7689    7694    7699    7704    7715    7720    7725    7730    7742    7790
                                7796
DT100    055062                 7452#   7824
DT107    055070                 7453#   7829    7855    7860    7900    7905
DT12     055030                 7447#   7477    7522    7558    7563    7568    7589    7669    7674    7679    7710    7748    7754
                                7760    7772    7778    7784    7802    7808    7814    7917    7923    7929
DT16     055042                 7449#
DT35     055050                 7450#   7625
DT5      055020                 7445#   7497    7502    7507    7512    7527    7532    7542    7547    7578    7583    7654    7659
                                7664    7736    7766    7834    7839    7845    7866    7872    7877    7883    7890    7895    7911
EAD      001230                  451#    696*    712*    744*    786*    835*   4949    5005    5103    5133
EMTVEC=  000030                  134#    583*    584*
EM1      041752                 6446#   7475
EM10     042460                 6503#   7510
EM101    051500                 7133#   7822
EM102    051566                 7142#   7827
EM103    051640                 7149#   7832
EM104    051733                 7159#   7837
EM105    052030                 7170#   7842
EM107    052165                 7187#   7853
EM11     042541                 6512#   7515
EM110    052242                 7195#   7858
EM111    052335                 7205#   7863
EM112    052437                 7217#   7869
EM113    052550                 7230#   7875
EM114    052631                 7239#   7880
EM115    052732                 7250#   7886
EM116    053100                 7268#   7893
EM117    053156                 7276#   7898
EM12     042623                 6521#   7520
EM120    053231                 7284#   7903
EM121    053277                 7291#   7908
EM122    053416                 7305#   7914
EM123    053532                 7318#   7920
EM124    053653                 7332#   7926
EM14     042676                 6529#   7530
EM16     042770                 6539#   7540
EM17     043033                 6545#   7545
EM2      042032                 6454#   7480
EM20     043114                 6554#   7550
EM21     043221                 6566#   7556
EM22     043316                 6577#   7561
EM23     043410                 6587#   7566
EM24     043501                 6597#   7571
EM25     043560                 6605#   7576
EM26     043633                 6613#   7581
EM27     043703                 6620#   7586    7592
EM3      042106                 6462#   7485
EM31     044073                 6641#   7598
```

```
MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 148
DQKKAA.P11    07-FEB-77 11:01               CROSS REFERENCE TABLE -- USER SYMBOLS

EM32     044235          6658#   7604
EM33     044365          6674#   7610
EM34     044514          6689#   7616
EM35     044637          6704#   7622
EM36     045000          6721#   7628
EM37     045157          6740#   7634
EM4      042167          6471#   7490
EM40     045337          6760#   7640
EM41     045462          6775#   7646
EM42     045606          6790#   7652
EM43     045676          6800#   7657
EM44     045745          6807#   7662
EM45     046023          6815#   7667
EM46     046102          6823#   7672
EM47     046173          6833#   7677
EM5      042251          6480#   7495
EM50     046260          6842#   7682
EM51     046346          6851#   7687
EM52     046433          6860#   7692
EM53     046521          6870#   7697
EM54     046614          6880#   7702
EM55     046710          6890#   7707
EM56     047027          6904#   7713
EM57     047114          6913#   7718
EM6      042330          6488#   7500    7525
EM60     047202          6922#   7723
EM61     047262          6930#   7728
EM62     047330          6937#   7733
EM63     047470          6953#   7739
EM64     047622          6968#   7745
EM65     047752          6983#   7751
EM66     050103          6998#   7757
EM67     050226          7013#   7763
EM7      042402          6495#   7505
EM70     050373          7031#   7769
EM71     050521          7046#   7775
EM72     050654          7062#   7781
EM73     051010          7078#   7787
EM74     051106          7089#   7793
EM75     051204          7100#   7799
EM76     051304          7111#   7805
EM77     051405          7122#   7811
EREG   = 177744           290#   1074    1673    1767    1782    1793    1864    1874    1912    1923    1965    1972    2301
                         2311    2436    2454    2462    2771    2781    2942    2960    3072    3084    3211    3229    3237
                         3387    3405    3562    3570    3732    3740    3887    3898    4032    4043    4167    4176    4547
                         4556    5208    5210
ERRVEC= 000004            127#    597     598*    609*   5571    5572*   5574*   5577*   5733    5734    5740*   5748*   5760*
                         5765*   5774*   5784*   5785*
FAKE     034224           712    5404#
GNS    = ****** U         344    6270    6271    6272    6273    6274    6277    6278    6279
HAD      033714          3803    4092    5341#
HIADD  = 000101           292#   1067
HMR    = 177752           285#   1114    1122    1126    1155    1215    1238    1244    1273    1284    1329    1333    1382
                         1392    1473    1482    2036    2050    2077    2114    2133    2246    2253    2383    2395    2517
                         2526    2553    2590    2604    2716    2723    2882    2891    3169    3181    3327    3336    3492
                         3506    3662    3676    3825    3828    3980    3983    4100    4105    4233    4283    4345    4614

MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 149
DQKKAA.P11    07-FEB-77 11:01               CROSS REFERENCE TABLE -- USER SYMBOLS

                         4671    4681    4722    4733    4941    4947    4986    4992    5095    5101    5114    5120
HMR0   = 000001           328#
HMR1   = 000002           329#   1163
HMR2   = 000004           330#   1170    1215    1238    1244    1273    1284    1329    1333    1382    1392    1473    1482
                         2036    2050    2077    2114    2133    2246    2253    2383    2395    2517    2526    2553    2590
                         2604    2716    2723    2882    2891    3169    3181    3327    3336    3492    3506    3662    3676
                         3825    3828    3980    3983    4100    4105    4233    4283    4345    4614    4671    4681    4722
                         4733    4941    4947    4986    4992    5095    5101    5114    5120
HMR3   = 000010           331#   1176
HMR4   = 000020           332#   1182
HMR5   = 000040           333#   1193
HRK05    034226           745    5411#
HPP03    034460           787    5459#
HT     = 000011            37#   5909    5950
HTU10    034714           833    5508#
HUBFN    034046           697     872    4975    5140    5358    5372#
HUBEO    034174           713    5197    5399#
IOTVEC= 000020            132#    581*    582*
IVEC     001226           450#    683*    874     876*   4977    5142    5144*   5360    5362*
KDPAR0= 172360            275#
KDPAR1= 172362            276#
KDPAR2= 172364            277#
KDPAR3= 172366            278#
KDPAR4= 172370            279#
KDPAR5= 172372            280#
KDPAR6= 172374            281#
KDPAR7= 172376            282#
KDPDR0= 172320            253#
KDPDR1= 172322            254#
KDPDR2= 172324            255#
KDPDR3= 172326            256#
KDPDR4= 172330            257#
KDPDR5= 172332            258#
KDPDR6= 172334            259#
KDPDR7= 172336            260#
KIPAR0= 172340            264#    962    4669*   4679*   5288    5334    5752
KIPAR1= 172342            265#
KIPAR2= 172344            266#
KIPAR3= 172346            267#   4328
KIPAR4= 172350            268#    914*    916     925*    926     933*    934     940*    941     989*    997*    999*   1008*
                         1014*   1021*   1027*   1036*   1044*   1052*   1054*   1429    1441    1443*   1446*   1447    1449
                         1451    1460    1470    1620*   1630    1647    1688    1697    1722*   1733*   1734    1739*   1774
                         2853*   3298*   3966*   3988*   4320*   4459    5059*   5060    5062*   5063    5105    5109
KIPAR5= 172352            269#   1437    1439    1453*   1456*   1470    1490    1493*   1495*   2867*   3312*   3965*   3989*
                         4460
KIPAR6= 172354            270#
KIPAR7= 172356            271#   5760
KIPDR0= 172300            242#
KIPDR1= 172302            243#
KIPDR2= 172304            244#
KIPDR3= 172306            245#
KIPDR4= 172310            246#    913*   1430    1627*   4324*   4350*
KIPDR5= 172312            247#
KIPDR6= 172314            248#   1433
KIPDR7= 172316            249#
LAST1    025746          3970    4063#
```

```
MD-11-DOKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 150
DOKKAA.P11    07-FEB-77 11:01           CROSS REFERENCE TABLE -- USER SYMBOLS

LF     = 000012         38#    335#   5944   5950   6342   6348   6356   6363   6368   6373   6378   6384   6391
                      6396   6403   6409   6414   6420   6427   6432   6438   7170   7205   7217   7230   7250
                      7291   7305   7318   7332
LOADD  = 000102        293#   1070   1676   1707   1765   2258   2434   2758   2927   3060   3209   3372   3543
                      3713   3878   4023   4158   4495   4538   4854   5202
LOC    = 000200        348#    356
MED    = 076600        297#    865    868   1066   1069   1080   1083   1550   1553   1586   1589   1665   1668
                      1675   1678   1683   1699   1706   1709   1751   1754   1759   1764   1769   1784   1795
                      1854   1857   1867   1876   1904   1907   1915   1926   1952   1955   1961   2063   2066
                      2099   2102   2151   2154   2275   2278   2282   2287   2290   2303   2313   2320   2421
                      2424   2428   2433   2440   2456   2464   2539   2542   2575   2578   2622   2625   2745
                      2748   2752   2757   2760   2773   2783   2790   2910   2913   2921   2926   2929   2952
                      2962   2969   3043   3046   3054   3059   3062   3074   3086   3094   3196   3199   3203
                      3208   3215   3231   3239   3355   3358   3366   3371   3374   3397   3407   3414   3526
                      3529   3537   3542   3545   3551   3564   3572   3583   3696   3699   3707   3712   3715
                      3721   3734   3742   3753   3815   3818   3862   3865   3872   3877   3880   3889   3900
                      3908   4007   4010   4017   4022   4025   4034   4045   4053   4142   4145   4152   4157
                      4160   4169   4178   4186   4225   4228   4275   4278   4337   4340   4489   4494   4499
                      4502   4522   4525   4532   4537   4540   4549   4550   4566   4606   4609   4719   4730
                      4799   4815   4818   4834   4837   4844   4848   4853   5196   5201   5204   5212   5223
                      5232   5240   5243   5260   5263   5794   5797
MMR0   = 177572        308#    861*   915*  1059*  1077*  1460*  1478*  1487*  1490*  1629*  1808*  2874*  2995*
                      3319*  3439*  3976*  4059*  4331*  4349*  4464*  4574*  5053*  5139*  5382*  5416*  5461*
                      5474*  5521*
MMR2   = 177576        309#
MMVEC  = 000250        143#
MSG1     040542        616    6327#
MSG10    041507        822    6414#
MSG11    041546        827    6420#
MSG12    041614        855    5464   6427#
MSG13    041645        753     850   5379   5419   5477   5524   6432#
MSG14    041705       5380    5420   5465   5478   5525   6438#
MSG2     040625       4785    6336#
MSG3     040670        635    6342#
MSG4     041202        642     671    733    775    807   6378#
MSG5     041242        660    6384#
MSG6     041312        670     708    725    767    799   6391#
MSG7     041414        728     770    802   6403#
MTBRC  = 172524        326#    843*  5511*  5530*
MTC    = 172522        325#    795    813*   816*   817*   830*   834    835    841*   842*   845*   853   5508*
                      5512*  5513*  5514*  5528*  5529*  5539*
MTCMA  = 172526        327#    844*  5531*
MTS    = 172520        324#    820    825    831    847   5516
NSSYN    004072        902    1065#
PAR      033634       2944    2948   2979   2990   3389   3393   3424   3435   5325#
PIRQ   = 177772         44#
PIRQVE = 000240        138#
PR0    = 000000         61#
PR1    = 000040         62#
PR2    = 000100         63#
PR3    = 000140         64#
PR4    = 000200         65#
PR5    = 000240         66#
PR6    = 000300         67#
PR7    = 000340         68#
PS     = 177776         41#     42
```

```
MD-11-DOKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 151
DOKKAA.P11    07-FEB-77 11:01           CROSS REFERENCE TABLE -- USER SYMBOLS

PSW    = 177776         42#   5364*
PVEC   = 000114        310#   1515*  1550*  1593*  1626*  1732*  1833*  1887*  1939*  2043*  2075*  2086*  2111*
                      2126*  2168*  2230*  2336*  2367*  2408*  2523*  2551*  2562*  2587*  2602*  2639*  2700*
                      2806*  2852*  2996*  3015*  3116*  3153*  3254*  3297*  3440*  3478*  3610*  3648*  3780*
                      3802*  3823*  3837*  3953*  4060*  4091*  4194*  4309*  4577*  4593*  4644*  4777*  4827*
PWRVEC = 000024        133#    587*   588*  4778*  4788*  4829*  6284*  6285*  6294*  6300*  6312*  6313*
QRK05    002232        652     720#
QRP03    002412        653     762#
QTU10    002540        654     794#
QUBEN    001764        650     660#
QUBEO    002156        651     704#
Q1       001726        639     642#   709    726    754    760    800
Q2       001704        632     635#   851    856
RDAT   = 000106        302#   1962
RDCHR  = 104406       6133    6277#
RDLIN  = 104407       6205    6278#
RDOCT  = 104410        636     661    730    772    804   6279#
RESVEC = 000010        128#
RJAM   = 000100        298#   2291   2761   2930   3063   3375   3546   3716   3881   4026   4161   4541   4845
                      5205
RKBA   = 177410        315#   5442*
RKCS   = 177404        313#    721    743    744    747*  5412   5431*  5433   5441*  5451*
RKDA   = 177412        316#    746*  5440*
RKDS   = 177400        311#    749   5425
RKER   = 177402        312#
RKWC   = 177406        314#   5439*
RLOG   = 000022        303#    866   1081   1551   1587   1666   1752   1855   1905   1953   2064   2100   2152
                      2276   2422   2540   2576   2623   2746   2911   3044   3197   3356   3527   3697   3816
                      3863   4008   4143   4226   4276   4338   4500   4523   4607   4816   4835   5241   5261
                      5795
RPBA   = 176720        300#    321#  5491*
RPCA   = 176722        322#    783*
RPCS   = 176714        319#    763    785    786   5459   5470   5490*  5500*
RPDA   = 176724        323#    784*
RPDS   = 176710        317#   5483
RPER   = 176712        318#
RPWC   = 176716        320#   5489*
RSER   = 000101        299#   1679   1710   1760   2283   2429   2753   2922   3055   3204   3367   3530   3708
                      3873   4018   4153   4490   4533   4849   5197
RTAG   = 000107        301#   1684   1700   1770   2304   2441   2774   2953   3216   3398   3584   3754   3892
                      4035   4187   4567   5224
SDPAR0 = 172260        231#
SDPAR1 = 172262        232#
SDPAR2 = 172264        233#
SDPAR3 = 172266        234#
SDPAR4 = 172270        235#
SDPAR5 = 172272        236#
SDPAR6 = 172274        237#
SDPAR7 = 172276        238#
SDPDR0 = 172220        209#
SDPDR1 = 172222        210#
SDPDR2 = 172224        211#
SDPDR3 = 172226        212#
SDPDR4 = 172230        213#
SDPDR5 = 172232        214#
SDPDR6 = 172234        215#
```

```
MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 152
DQKKAA.P11    07-FEB-77 11:01           CROSS REFERENCE TABLE -- USER SYMBOLS

SDPDR7= 172236          216#
SETUP   001232          452#    697*    713*    745*    787*    833*    872     4919    4975    5073    5107    5140    5358
SIPAR0= 172240          220#
SIPAR1= 172242          221#
SIPAR2= 172244          222#
SIPAR3= 172246          223#
SIPAR4= 172250          224#
SIPAR5= 172252          225#
SIPAR6= 172254          226#
SIPAR7= 172256          227#
SIPDR0= 172200          198#
SIPDR1= 172202          199#
SIPDR2= 172204          200#
SIPDR3= 172206          201#
SIPDR4= 172210          202#
SIPDR5= 172212          203#
SIPDR6= 172214          204#
SIPDR7= 172216          205#
SKTST   001234          453#    898*    1106*   1144*   1209*   1235*   1268*   1326*   1369*   1423*   1514*   1620*   1832*
                        2028*   2229*   2366*   2509*   2699*   2846*   3014*   3152*   3291*   3477*   3647*   3801*   3949*
                        4090*   4216*   4264*   4318*   4388*   4592*   4660*   4699*   4774*   4908*   5041*   5247
SR0   = 177572          147#    308     5749    5764*   5771*
SR1   = 177574          148#
SR2   = 177576          149#    309
SR3   = 172516          150#    5761*
STACK = 001100           32#    579     862     982     2169    4793
START   001362          355     572#
START1  003056          633     698     714     756     788     854     859#    5182
STKLMT= 177774           43#
SWEEP   035134          1533    1545    1568    1577    1642    1660    1729    1746    1982    2165    2270    2636    2740
                        3834    4620    4641    5546#
SWR     001134          417#    577     599*    601     607*    614*    626     628     631     899     1424    1621    2847
                        3292    3950    4319    4450    4775    4790    4909    5042    5045    5254    5377    5417    5462
                        5475    5522    5566    5584    5591    5629    5636    5648    5651    6292    6305*
SWREG   000176          346#    607
SW0   = 000001           96#
SW00  = 000001           86#    96
SW01  = 000002           85#    95
SW02  = 000004           84#    94
SW03  = 000010           83#    93
SW04  = 000020           82#    92
SW05  = 000040           81#    91
SW06  = 000100           80#    90
SW07  = 000200           79#    89      628     4775
SW08  = 000400           78#    88      626     631     4909    5042
SW09  = 001000           77#    87      5254
SW1   = 000002           95#
SW10  = 002000           76#
SW11  = 004000           75#
SW12  = 010000           74#    899     1424    1621    2847    3292    3950    4319    4450    5045
SW13  = 020000           73#    5377    5417    5462    5475    5522
SW14  = 040000           72#
SW15  = 100000           71#
SW2   = 000004           94#
SW3   = 000010           93#
SW4   = 000020           92#

MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 153
DQKKAA.P11    07-FEB-77 11:01           CROSS REFERENCE TABLE -- USER SYMBOLS

SW5   = 000040           91#
SW6   = 000100           90#
SW7   = 000200           89#
SW8   = 000400           88#
SW9   = 001000           87#
TAB     001752          648     650#
TAD1    026442          4093    4097*   4099    4115    4203#
TAD2    025242          3804    3824    3849    3915#
TAG     033606          1648    1689    1698    1775    2950    2981    3395    3426    5313#
TBITVE= 000014          129#
TKVEC = 000060          136#
TPAT    026432          4097    4098    4102    4107    4109    4127    4132    4198#
TPVEC = 000064          137#
TRAPVE= 000034          135#    585*    586*    5737    5738*   5743*
TRTVEC= 000014          130#
TST1    003162          896#

TST10   006000          1326    1347    1367#
TST11   006166          1369    1390    1393    1421#
TST12   006626          1423    1426    1479    1480    1512#
TST13   007140          1514    1594    1618#
TST14   010230          1620    1623    1830#
TST15   012000          1832    1986    2026#
TST16   012734          2028    2170    2227#
TST17   013406          2229    2364#
TST2    004164          898     901     1061    1103#
TST20   014100          2366    2410    2507#
TST21   015000          2509    2640    2697#
TST22   016000          2699    2808    2844#
TST23   016646          2846    2849    3012#
TST24   020000          3014    3118    3150#
TST25   020456          3152    3289#
TST26   022000          3291    3294    3442    3475#
TST27   024000          3477    3612    3645#
TST3    004320          1106    1142#
TST30   024566          3647    3799#
TST31   025244          3801    3838    3947#
TST32   025750          3949    3952    4061    4088#
TST33   026444          4090    4195    4214#
TST34   026634          4216    4238    4262#
TST35   027030          4264    4290    4316#
TST36   027300          4318    4320    4352    4386#
TST37   030260          4388    4452    4590#
TST4    004616          1144    1194    1207#
TST40   030524          4592    4658#
TST41   030674          4660    4672    4676    4697#
TST42   031132          4699    4726    4743    4772#
TST43   031524          4774    4776    4829    4906#
TST44   032240          4908    4911    4976    4980    5039#
TST5    004712          1209    1216    1233#
TST6    005044          1235    1245    1249    1266#
TST7    005364          1268    1279    1290    1306    1324#
TYPOS = 104405          5171    6274#
TYPE  = 104401          616     635     642     660     670     671     708     725     728     733     753     767     770
                        775     799     802     807     822     827     850     855     4785    5169    5172    5379    5399
                        5419    5420    5464    5465    5477    5478    5524    5525    5631    5639    5673    5690    5692
                        5695    5697    5701    5708    5862    5914    6067    6140    6146    6151    6155    6160    6161
```

```
                              6163    6166    6170    6234    6236    6270#   6314
TYPOC = 104402                5681    5785    6271#
TYPON = 104404                6273#
TYPOS = 104403                6272#
T01L01  004242                1109    1114#
T01L02  004264                1115    1120#
T01L03  004252                1116#   1123    1127
T02L01  004404                1148    1155#
T02L02  004462                1164    1170#
T02L03  004504                1171    1176#
T02L04  004526                1177    1182#
T02L05  004550                1183    1188#
T02L06  004454                1167#   1174    1180    1186    1190
T04L01  005144                1274    1281#
T04L02  005220                1285    1292#
T04L03  005300                1293    1303#
T05L01  005472                1330    1341#
T05L02  005514                1334    1339    1346#
T06L01  006260                1437#   1494    1496
T06L02  006366                1440    1454    1457#
T06L03  006276                1438    1441#   1491
T06L04  006620                1448    1450    1498#
T06L05  006316                1442    1446#
T06L06  006324                1444    1447#
T06L07  006360                1452    1456#
T06L08  006440                1469#   1476    1485
T06L09  006522                1471    1481#
T06L10  006566                1474    1483    1490#
T06L11  006610                1492    1495#
T06L12  006266                1435    1439#
T07L01  007354                1626    1656#
T07L02  007650                1632    1725#
T07L03  007254                1635#
T07L04  007232                1630#   1650    1691    1715    1723
T07L05  010214                1654    1695    1720    1735    1780    1791    1802    1808#
T07L06  007520                1674    1697#
T07L07  007636                1704    1719    1722#
T07L08  007734                1732    1742#
T07L09  007674                1734#   1740    1777    1788    1799    1805
T07L10  010076                1760    1782#
T07L11  010136                1783    1793#
T07L12  010176                1794    1804#
T08L01  010350                1833    1851#
T08L02  011052                1849    1872    1881    1899    1921    1932    1945    1970    1978#
T08L03  010432                1863    1874#
T08L04  010460                1865    1883#
T08L05  010466                1875    1887#
T08L06  010262                1836#   1895
T08L07  010546                1887    1901#
T08L09  010630                1913    1923#
T08L10  010504                1890#   1918    1929    1937
T08L11  010662                1924    1934#
T08L12  010676                1935    1939#
T08L13  010740                1939    1949#
T08L14  010706                1941#   1947    1967    1974
T08L15  011032                1966    1972#
```

```
T08L16  010724                1944#   1973
T09L01  006770                1515    1541#
T09L02  006722                1519    1526#
T09L03  007072                1558    1573#
T09L04  007056                1564#
T09L06  007110                1539    1571    1583#
T11H01  015172                2700    2736#
T11H02  015040                2704#   2734
T11H03  015100                2712    2715#   2719
T11H04  015400                2717    2796#
T11H05  015420                2724    2800#
T11H06  015444                2729    2768    2779    2787    2794    2806#
T11H07  015304                2763    2770#
T11H08  015340                2772    2781#
T11H09  015364                2782    2789#
T11H10  015436                2799    2803#
T11H11  015122                2722#   2726
T11H12  015150                2731#   2732
T11L01  013126                2230    2266#
T11L02  012774                2234#   2264
T11L03  013034                2242    2245#   2249
T11L04  013334                2247    2326#
T11L05  013354                2254    2330#
T11L06  013400                2259    2298    2309    2317    2324    2336#
T11L07  013240                2293    2300#
T11L08  013274                2302    2311#
T11L09  013320                2312    2319#
T11L10  013372                2329    2333#
T11L11  013056                2252#   2256
T11L12  013104                2261#   2262
T12H01  020200                3153    3193#
T12H02  020030                3156#   3191
T12H06  020130                3177    3179#
T12H07  020430                3170    3182    3249#
T12H08  020450                3187    3272    3235    3243    3254#
T12H09  020322                3212    3224#
T12H11  020342                3227    3229#
T12H12  020366                3230    3237#
T12H13  020160                3188#   3190
T12H14  020412                3238    3245#
T12L01  013646                2367    2418#
T12L02  013436                2378#   2405
T12L06  013536                2391    2393#
T12L07  013624                2384    2396    2412#
T12L08  013606                2401    2408#   2416    2447    2460    2468    2473
T12L09  013770                2437    2449#
T12L11  014010                2452    2454#
T12L12  014034                2455    2462#
T12L13  013566                2402#   2404
T12L14  014060                2463    2470#
T13H01  020744                3297    3352#
T13H02  020602                3317#   3350
T13H03  020634                3326#   3331
T13H04  021216                3329    3420#
T13H05  020664                3335#   3341
T13H06  021264                3338    3432#
```

```
T13H07  021312              3345    3380    3403    3411    3418    3430    3439#
T13H08  021056              3378    3382#
T13H09  021072              3384    3387#
T13H10  021156              3388    3405#
T13H11  021122              3392    3395#
T13H12  021202              3406    3413#
T13H13  021246              3426#   3437
T13H15  020732              3347    3348#
T13L01  016766              2852    2907#
T13L02  016124              2872#   2905
T13L03  016156              2881#   2886
T13L04  016540              2884    2975#
T13L05  016206              2890#   2896
T13L06  016606              2893    2987#
T13L07  016634              2900    2935    2958    2966    2973    2985    2995#
T13L08  016400              2933    2937#
T13L09  016414              2939    2942#
T13L10  016500              2943    2960#
T13L11  016444              2947    2950#
T13L12  016524              2961    2968#
T13L13  016570              2981#   2992
T13L15  016254              2902    2903#
T14L01  016760              3015    3040#
T14L02  016704              3019#   3071    3112
T14L03  017242              3022    3106#
T14L04  016746              3024    3034#
T14L05  016734              3026    3030#
T14L06  016712              3020#   3028    3032    3038
T14L07  017304              3035    3068    3080    3116#
T14L08  017072              3065    3070#
T14L09  017150              3073    3084#
T14L10  016720              3023#   3082
T14L11  017176              3085    3093#
T14L12  017130              3079#   3091    3104    3114
T15H01  024210              3648    3693#
T15H02  024466              3664    3678    3761#
T15H03  024510              3667    3767#
T15H04  024560              3687    3725    3738    3746    3759    3765    3780#
T15H05  024034              3651#   3691
T15H06  024334              3720    3727#
T15H08  024350              3729    3732#
T15H12  024170              3686    3688#   3690
T15H13  024374              3733    3740#
T15H14  024420              3741    3748#
T15H15  024526              3681    3772#
T15H16  024542              3770    3775#
T15H17  024112              3666    3668#
T15H18  024154              3680    3682#
T15H21  024064              3661#   3669
T15H22  024126              3675#   3683
T15L01  022210              3478    3523#
T15L02  022166              3494    3508    3591#
T15L03  022510              3497    3597#
T15L04  022560              3517    3555    3568    3576    3589    3595    3610#
T15L05  022034              3481#   3521
T15L06  022334              3550    3557#
```

```
T15L08  022350              3559    3562#
T15L12  022170              3516    3518#   3520
T15L13  022374              3563    3570#
T15L14  022420              3571    3578#
T15L15  022526              3511    3602#
T15L16  022542              3600    3605#
T15L17  022112              3496    3498#
T15L18  022154              3510    3512#
T15L21  022064              3491#   3499
T15L22  022126              3505#   3513
T16L01  025534              3953    4004#
T16L02  025504              3981    3996#
T16L03  025470              3984    3992#
T16L04  025732              3986    4002    4030    4041    4050    4059#
T16L05  025410              3977#   3990    3999
T16L06  025516              3994    3998#
T16L07  025642              4028    4032#
T16L08  025674              4033    4043#
T16L09  025720              4044    4052#
T17L01  026224              4091    4139#
T17L02  026104              4101    4114#
T17L03  026174              4103    4131#
T17L04  026136              4106    4122#
T17L05  026152              4108    4126#
T17L06  026422              4110    4120    4137    4165    4174    4183    4194#
T17L07  026016              4097#   4112
T17L08  026010              4096#   4116
T17L09  026120              4116#   4124
T17L10  026210              4129    4133#
T17L11  026332              4163    4167#
T17L12  026356              4168    4176#
T17L13  026402              4177    4185#
T18L01  030036              4389    4519#
T18L02  027710              4390    4485#
T18L03  030004              4438    4444    4510#
T18L05  027564              4455    4456#
T18L06  027620              4463#   4470
T18L07  027630              4465#   4469
T18L09  027650              4467    4472#   4481    4483
T18L10  030234              4475    4508    4517    4545    4554    4563    4574#
T18L11  027340              4391    4392#
T18L12  030144              4543    4547#
T18L13  030170              4548    4556#
T18L14  030214              4557    4565#
T19L01  030610              4662    4669#
T19L02  030604              4665    4668#
T19L03  030650              4667    4670#
T19L04  030630              4673#   4682
T20L01  031412              4777    4831#
T20L02  031226              4778    4788#
T20L03  031246              4788    4793#
T20L04  031334              4809    4812#
T20L05  031520              4847    4859#
T20L06  031374              4827#   4857    4860
T21L01  031712              4932    4941#
T21L02  031704              4935    4939#
```

```
MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 158
DQKKAA.P11     07-FEB-77 11:01            CROSS REFERENCE TABLE -- USER SYMBOLS

T21L03  032112            4938     4984#
T21L04  032152            4942     4987     4996#
T21L05  032200            4948     4950     4993     5002#
T21L07  032024            4952     4966#
T21L08  032000            4954     4960#
T21L09  031614            4919#    4958     4964     4973     4996     5002
T21L10  032060            4967     4975#    5000     5008     5010
T21L11  031742            4949#    4994
T22L01  032534            5086     5095#
T22L02  032754            5064     5106     5111     5120     5136     5139#
T22L03  032526            5089     5094#
T22L04  032626            5092     5113#
T22L05  032670            5096     5115     5124#
T22L06  032716            5102     5104     5121     5130#
T22L08  032350            5058#    5108     5110
T22L10  032566            5103#    5122
T22L11  032436            5073#    5124     5130
T23L01  030504            4615     4624     4630     4637#
T24H01  014670            2519     2642#
T24H02  014242            2523     2536#
T24H03  014712            2520     2640#
T24H04  014734            2534     2655#
T24H05  014200            2525#    2549
T24H06  014756            2555     2594     2661#
T24H07  014404            2562     2572#
T24H08  014346            2564#    2569     2585
T24H09  014504            2591     2596#
T24H10  014334            2562#    2593     2607     2615
T24H12  014450            2580#    2590
T24H13  014552            2605     2610#
T24H14  014720            2600     2649#
T24H15  014742            2570     2616     2658#
T24H16  014610            2602     2618#
T24H17  014522            2603#    2630
T24H18  014646            2632#    2646     2653     2659     2666
T24H19  014156            2518     2520#
T24H20  014140            2516#    2521
T24H21  014172            2524#
T24H22  014216            2527     2530#
T24L01  012624            2038     2172#
T24L02  012154            2043     2060#
T24L03  012646            2052     2178#
T24L04  012670            2058     2185#
T24L05  012076            2045#    2073
T24L06  012712            2079     2110     2191#
T24L07  012316            2086     2096#
T24L08  012260            2080#    2093     2109
T24L09  012416            2115     2120#
T24L10  012246            2086#    2117     2136     2144
T24L12  012362            2112#    2122
T24L13  012500            2128     2130     2134     2139#
T24L14  012654            2137     2179#
T24L15  012676            2094     2145     2186#
T24L16  012536            2126     2147#
T24L17  012434            2127#    2159
T24L18  012574            2161#    2176     2183     2189     2196


MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 159
DQKKAA.P11     07-FEB-77 11:01            CROSS REFERENCE TABLE -- USER SYMBOLS

T24L19  012062            2037     2039#
T24L20  012044            2035#    2040
T24L22  012130            2046     2048     2051     2054#
T25L01  031036            4714     4728#
T25L02  031064            4723     4734     4737#
T25L03  031030            4724#    4735     4745
T25L04  030730            4706#    4725     4740
T27L01  026510            4218     4227#
T27L02  026564            4234     4240#
T27L03  026610            4236     4246#
T27L04  026554            4237#    4244     4250
T28L01  026706            4266     4272#
T28L02  026772            4284     4292#
T28L03  027016            4286     4298#
T28L04  026752            4287#    4296     4300
T29L01  027152            4323     4334#
T29L02  027242            4346     4354#
T29L03  027266            4348     4360#
T29L04  027216            4349#    4358     4362
T30L01  024646            3802     3812#
T30L02  025020            3810     3853#
T30L03  024774            3826     3847#
T30L04  024752            3829     3840#
T30L05  024730            3830#    3845     3851     3857     3885     3896     3905     3913
T30L06  025042            3823     3859#
UBEAPT  002050             627      629      678#
UDPAR0= 177660             187#
UDPAR1= 177662             188#
UDPAR2= 177664             189#
UDPAR3= 177666             190#
UDPAR4= 177670             191#
UDPAR5= 177672             192#
UDPAR6= 177674             193#
UDPAR7= 177676             194#
UDPDR0= 177620             165#
UDPDR1= 177622             166#
UDPDR2= 177624             167#
UDPDR3= 177626             168#
UDPDR4= 177630             169#
UDPDR5= 177632             170#
UDPDR6= 177634             171#
UDPDR7= 177636             172#
UIPAR0= 177640             176#
UIPAR1= 177642             177#
UIPAR2= 177644             178#
UIPAR3= 177646             179#
UIPAR4= 177650             180#
UIPAR5= 177652             181#
UIPAR6= 177654             182#
UIPAR7= 177656             183#
UIPDR0= 177600             154#
UIPDR1= 177602             155#
UIPDR2= 177604             156#
UIPDR3= 177606             157#
UIPDR4= 177610             158#
UIPDR5= 177612             159#
```

```
MD-11-DOKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 160
DOKKAA.P11     07-FEB-77 11:01          CROSS REFERENCE TABLE -- USER SYMBOLS

UIPOR6= 177614           160#
UIPOR7= 177616           161#
UPERR   033142           860    1593    1809    1985    2075    2111    2160    2336    2408    2551    2587    2619    2806
                        2996    3116    3254    3440    3610    3700    3837    4060    4194    4577    4644    4827    5192#
UPR     033326          5216    5220    5229    5237#
UP1     033260          5207    5218#
UP2     033272          5209    5222#
UP3     033314          5211    5231#
UT4     033352           859    1060    1076    4237    4287    5251#
VEC     034000          4913    5065    5358#
VIP     033434          1459    1646    2977    2989    3422    3434    3998    5070    5273#
WFLT  = 000304           305#
WINIT = 000352           307#   4720    4731
WLOG  = 000222           304#    869    1084    1554    1590    1669    1755    1858    1908    1956    2067    2103    2155
                        2279    2425    2543    2579    2626    2749    2914    3047    3200    3359    3530    3700    3819
                        3866    4011    4146    4229    4279    4341    4503    4526    4610    4819    4838    5244    5264
                        5798
WSW   = 000226           306#   4800
$APTHD  001000           367     373#
$ASTAT= ****** U        5984    5999
$ATYC   037056          5955    5957#
$ATY1   037032          5953#
$ATY3   037040          5899    5954#
$ATY4   037050          5644    5956#
$BASE   001312           499#
$BDADR  001122           412#
$BDDAT  001126           414#
$BELL   035610          5631    5664#
$CDW1   001316           501#
$CDW2   001320           502#
$CHARC  037026          5916*   5926*   5933    5942*   5947#
$CKSWR= ****** U        6277
$CMTAG  001100           400#    574     575     583     589     590     591
$CNTLG  040117          6181#
$CNTLU  040112          6155    6180#
$CORE   036204          5747    5774#
$CPUOP  001264           473#
$CRLF   001207           442#   5639    5665    5673    5692    5697    5701    5915    5950    6160    6180    6239
$CROUT  036234          5774    5781#
$DBLK   036540          5828    5862    5870#
$DDW0   001322           503#
$DDW1   001324           504#
$DDW10  001346           513#
$DDW11  001350           514#
$DDW12  001352           515#
$DDW13  001354           516#
$DDW14  001356           517#
$DDW15  001360           518#
$DDW2   001326           505#
$DDW3   001330           506#
$DDW4   001332           507#
$DDW5   001334           508#
$DDW6   001336           509#
$DDW7   001340           510#
$DDW8   001342           511#
$DDW9   001344           512#


MD-11-DOKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 161
DOKKAA.P11     07-FEB-77 11:01          CROSS REFERENCE TABLE -- USER SYMBOLS

$DEVCT  001246           464#
$DEVM   001314           500#
$DOAGN  033116          5165    5174    5180#
$DTBL   036530          5831    5866#
$ENDAD  033106           386    5176#   5658
$ENDCT  033054           589    5167#
$ENDMG  033125          5169    5184#
$ENULL  033122          5172    5183#
$ENV    001256           469#    624    4780    4794    5641    5894    5962    5986
$ENVM   001257           470#    612    5896    5901    5964
$EOP    033020          1112    1118    1153    1160    1370    5041    5044    5047    5141    5157#   5383    5422    5467
                        5480    5526
$EOPCT  033046           589*   5164#   5168
$ERFLG  001103           403#   1710    3079    4742    5557    5580    5582    5588*   5611    5626*   5663
$ERMAX  001115           409#    592*   5582    5605*   5611
$ERROR  035412           583    5625#
$ERRPC  001116           410#   5633*   5634*   5635    5664    5679    7443    7445    7447    7449    7450    7452    7453
$ERRTB  055074          5687    7472#
$ERRTY  035614          5638    5672#
$ERTTL  001112           407#   5632*   5663
$ESCAP  035606           591*   5604*   5654    5656    5663#
$ETABL  001256           468#
$ETEND  001362           379     521#
$FATAL  001240           461#   5990*
$FFLG   037276          5953*   5956*   5984    5993*   6001#
$FILLC  001152           425#   5919    5950
$FILLS  001151           424#   5950
$GDADR  001120           411#
$GDDAT  001124           413#
$GET42  033076          5173#
$GTSWR= ****** U        6276
$HD   = 000001            12      13
$HIBTS  001000           374#
$HIOCT  040304          6227*   6238#
$ICNT   001104           404#   5595*   5596    5598*   5610
$ILLUP  040524          6294    6300    6317#
$ITEMB  001114           408#   5635*   5643    5665    5676
$KTNEX  036176          5748    5773#
$KTOUT  036166          5765    5770#
$KT11   036034           906*   1427*   1624*   2850*   3295*   3954*   4321*   4453*   5049*   5746*   5750*   5771*
$LF     001210           443#   5665    5950    6170    6180    6239
$LFLG   037275          5994*   6000#
$LPADR  001106           405#    593*   5586*   5602*   5607    5609
$LPERR  001110           406#    594*   1476*   1485*   1650*   1691*   1715*   1777*   1788*   1799*   1805*   1918*   1929*
                        1967*   1974*   2093*   2117*   2136*   2144*   2262*   2404*   2569*   2593*   2607*   2615*   2732*
                        2902*   3071*   3112*   3190*   3347*   3520*   3690*   3999*   4116*   4391*   4455*   4740*   4996*
                        5002*   5124*   5130*   5586    5603*   5609    5653
$LSTAD  036320          5786*   5802#
$LSTBK  036322           900     916     934     991     993    1004    1012    1019    1033    1041    1439    1449    1630
                        1734    2853    2857    3298    3302    3956    3966    4461    5051*   5063    5787*   5803#
$MADR1  001270           486#
$MADR2  001274           490#
$MADR3  001300           493#
$MADR4  001304           496#
$MAIL   001236           375     379     459#    611    5601    5641    5894
$MAMS1  001266           480#
```

```
MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 162
DQKKAA.P11     07-FEB-77 11:01          CROSS REFERENCE TABLE -- USER SYMBOLS

$MAMS2  001272              488#
$MAMS3  001276              491#
$MAMS4  001302              494#
$MBADR  001002              375#
$MFLG   037274             5954*   5960    5995*   5999#
$MNEW   040135             6184#
$MSGAD  001252              466#   5970*   5973
$MSGLG  001254              467#   5975*
$MSGTY  001236              460#   5968    5976*   5988    5992*
$MSWR   040124             6182#
$MTYP1  001267              481#
$MTYP2  001273              489#
$MTYP3  001277              492#
$MTYP4  001303              495#
$MXCNT  035410             5599    5610#
$NULL   001150              423#   5921    5950
$NWTST= 000001              880#    882    1089#   1091    1129#   1131    1200#   1202    1223#   1225    1256#   1258    1315#
                           1317    1353#   1355    1399#   1401    1500#   1502    1597#   1599    1811#   1813    1993#   1995
                           2198#   2200    2338#   2340    2477#   2479    2668#   2670    2813#   2815    2998#   3000    3124#
                           3126    3258#   3260    3449#   3451    3619#   3621    3782#   3784    3917#   3919    4065#   4067
                           4205#   4207    4252#   4254    4302#   4304    4364#   4366    4580#   4582    4646#   4648    4684#
                           4686    4747#   4749    4882#   4884    5012#   5014
$OCNT   037522             6039*   6068*   6081#
$OMODE  037524             6034*   6038*   6043    6046*   6057*   6083#
$OVER   035372             5567    5587    5597    5606#
$PASS   001244              463#    611*   5161*   5162*   5170    5183    5593    5611
$PASTM  001006              377#
$POWER  040532             6315    6320#
$PWRDN  040364              587    4828    6284#   6312
$PWRMG  040520             6315#
$PWRUP  040436             6294    6300#
$QUES   001206              441#   5665    5950    6163    6180    6236    6239
$RDCHR  037526             6100#   6277
$RDDEC= ****** U           6280
$RDLIN  037646             6128#   6278
$RDOCT  040146             6280#   6279
$RDSZ = 000010             6121#
$REGAD  001154              427#
$REG0   001156              429#
$REG1   001160              430#    980*   1068*   1107*   1108    1114*   1122*   1126*   1150*   1165*   1172*   1178*   1184*
                           1196*   1217*   1252*   1276*   1287*   1296*   1309*   1336*   1342*   1375*   1385*   1394*   1521*
                           1536*   1682*   1713*   1763*   1837*   2173*   2179*   2186*   2192*   2286*   2333*   2414*   2432*
                           2643*   2649*   2656*   2662*   2756*   2803*   2925*   3058*   3108*   3207*   3251*   3370*   3541*
                           3592*   3605*   3711*   3762*   3775*   3841*   3848*   3854*   3876*   4021*   4117*   4134*   4156*
                           4241*   4247*   4293*   4355*   4356*   4493*   4511*   4536*   4626*   4633*   4673*   4738*   4808*
                           4852*   4997*   5003*   5071    5125*   5131*   5200*   5253*   5291*   5293*   7443    7445    7447
                           7449    7450    7452
$REG2   001162              431#    979*   1071*   1110*   1116*   1151*   1166*   1173*   1179*   1185*   1197*   1218*   1253*
                           1277*   1288*   1297*   1310*   1337*   1343*   1376*   1386*   1395*   1522*   1537*   1677*   1708*
                           1766*   1838*   1888*   2174*   2180*   2187*   2193*   2289*   2328*   2331*   2413*   2435*   2644*
                           2650*   2657*   2663*   2759*   2798*   2801*   2928*   3061*   3109*   3210*   3250*   3373*   3544*
                           3593*   3606*   3714*   3763*   3776*   3842*   3849*   3855*   3879*   4024*   4115*   4123*   4128*
                           4159*   4242*   4248*   4294*   4496*   4512*   4539*   4628*   4634*   4674*   4739*   4855*   4998*
                           5004*   5072    5126*   5132*   5203*   5252*   5281*   5282*   5290*   5291    5298*   5304*   7443
                           7445    7447    7450
$REG3   001164              432#    945*   1074*   1219*   1298*   1311*   1396*   1466*   1649*   1687*   1705*   1773*   1786*

MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 163
DQKKAA.P11     07-FEB-77 11:01          CROSS REFERENCE TABLE -- USER SYMBOLS

                           1797*   1804*   1869*   1878*   1917*   1928*   1963*   2294*   2307*   2315*   2322*   2444*   2458*
                           2466*   2470*   2471*   2764*   2777*   2785*   2792*   2917*   2918*   2956*   2964*   2971*   2982*
                           3050*   3051*   3076*   3088*   3096*   3110*   3219*   3233*   3241*   3245*   3246*   3362*   3363*
                           3401*   3409*   3416*   3427*   3533*   3534*   3553*   3566*   3574*   3587*   3607*   3703*   3704*
                           3723*   3736*   3744*   3757*   3777*   3869*   3878*   3893*   3902*   3910*   4014*   4015*   4038*
                           4047*   4055*   4133*   4149*   4150*   4171*   4180*   4190*   4486*   4487*   4513*   4529*   4530*
                           4551*   4560*   4570*   4841*   4842*   5214*   5218*   5227*   5234*   7443    7447
$REG4   001166              433#    981*   1299*   1312*   1690*   1714*   1776*   1787*   1798*   1836*   1890*   1960*   1964
                           2295*   2438*   2439*   2450*   2453*   2765*   2937*   2940*   2951*   3070*   3098*   3100*   3102*
                           3111*   3213*   3214*   3225*   3228*   3382*   3385*   3396*   3557*   3560*   3578*   3580*   3582*
                           3599*   3604*   3727*   3730*   3748*   3750*   3752*   3769*   3774*   4127*   4132*   4514*   5193*
                           5194*   5210    7443    7450
$REG5   001170              434#
$PTNAD  033120             5182#
$R2A  = ****** U           6280
$SAVRE= ****** U           6280

$SAVR6  040530             6293*   6301    6302*   6303*   6319#
$SCOPE  035152              581    5565#
$SETUP= 000037              337#    580     581     583     585     587     589     590     591     593    5159    5566    5626
                           5651    5658    6089    6186
$SIZE   035750              907    1428    1625    2851    3296    3955    4322    4454    5050    5729#
$SIZEX  036240             5772    5782#
$STUP = 177777              337#
$SVLAD  035136             5575    5600#
$SVPC = 001014              384#    389
$SWR  = 167000                2#     12      17      18      19      20      21      22      23      24     590     591     593
                            594     897    1104    1143    1200    1234    1267    1325    1368    1422    1513    1619    1831
                           2027    2220    2365    2506    2698    2845    3013    3151    3290    3476    3646    3800    3940
                           4089    4215    4263    4317    4387    4591    4659    4698    4773    4907    5040    5154    5160
                           5175    5181    5183    5558    5559    5560    5561    5562    5566    5578    5580    5581    5582
                           5589    5590    5591    5603    5606    5609    5617    5618    5619    5620    5621    5629    5636
                           5648    5651    5663    5664    6316
$SWREG  001260              471#    614
$S=RNK= 000000             5562
$TESTN  001242              462#   5601*
$TIMES  035406              590*   5160*   5589*   5596    5599*   5609#
$TKB    001142              420#   6087    6104    6110
$TKS    001140              419#   6097    6102    6108
$TMP0   001172              435#    874*    875*    876     877*   1458*   1645*   1647*   1649    1688*   1690    1697*   1703
                           1714    1774*   1776    2951    2976*   2982    2988*   3396    3421*   3427    3433*   3805    3993*
                           3997*   4094    4662*   4663*   4664    4666    4706*   4708*   4710    4713    4790*   4798    4930*
                           4931*   4934    4937    4977*   4978*   4979*   5069*   5084*   5085*   5088    5091    5142*   5143*
                           5144    5145*   5275    5302    5315*   5317*   5325*   5328*   5331*   5333*   5334*   5335*   5341*
                           5343*   5344*   5345    5347*   5350*   5351*   5360*   5361*   5362    5363*
$TMP1   001174              436#   4707*   4709*   4710    4712*   4713    4932*   4933*   4934    4936*   4937    5086*   5087*
                           5088    5090*   5091
$TMP2   001176              437#
$TMP3   001200              438#
$TMP4   001202              439#   5446*   5448*   5449    5493*   5496*   5500    5533*   5535*   5539
$TMP5   001204              440#   5372*   5375*   5411*   5414*   5424*   5427*   5432*   5435*   5444*   5447*   5451    5464*
                           5472*   5482*   5485*   5495*   5497*   5498    5515*   5518*   5534*   5536*   5537
$TN   = 000045               12#    880     897#    898    1061    1089    1104#   1106    1129    1143#   1144    1194    1200
                           1208#   1209    1216    1223    1234#   1235    1245    1249    1256    1267#   1268    1279    1290
                           1306    1315    1325#   1326    1347    1353    1368#   1369    1388    1393    1399    1422#   1423
                           1426    1479    1488    1500    1513#   1514    1594    1597    1619#   1620    1623    1811    1831#
                           1832    1986    1993    2027#   2028    2170    2198    2220#   2229    2330    2365#   2366    2410
```

```
MD-11-DOKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 164
DOKKAA,P11     07-FEB-77 11:01               CROSS REFERENCE TABLE -- USER SYMBOLS

                              2477    2500#   2509    2640    2668    2698#   2699    2800    2813    2845#   2846    2849    2998
                              3013#   3014    3118    3124    3151#   3152    3258    3290#   3291    3294    3442    3449    3476#
                              3477    3612    3619    3646#   3647    3782    3800#   3801    3838    3917    3948#   3949    3952
                              4061    4065    4089#   4090    4195    4205    4215#   4216    4238    4252    4263#   4264    4290
                              4302    4317#   4318    4320    4352    4364    4387#   4388    4452    4580    4591#   4592    4646
                              4659#   4660    4672    4676    4684    4698#   4699    4726    4743    4747    4773#   4774    4776
                              4829    4882    4907#   4908    4911    4976    4980    5012    5840#
$TPB     001146                422#   5939#   5950
$TPFLG   001153                426#   5888    5950
$TPS     001144                421#   5937    5950
$TRAP    040306                585    6247#
$TRAP2   040330               6258#   6269
$TRP   = 000011               6262#   6271#   6272#   6273#   6274#   6275#   6277    6278#   6279#   6280#
$TRPAD   040342               6252    6269#
$TSTM    001004                376#
$TSTNM   001102                402#   5159#   5557    5600#   5601    5606    5611    5628    5663
$TTYIN   040102               6130    6131    6143    6161    6175    6179#
$TYPBN=  ****** U             6275
$TYPDS   036324               5816#   6274
$TYPE    036550               5888#   5901    6262    6270
$TYPEC   036762               5918    5925    5932    5937#   5938
$TYPEX   037030               5943    5945    5948#
$TYPOC   037324               6037#   6271
$TYPON   037340               6036    6039#   6273
$TYPOS   037300               6032#   6272
$UNIT    001250                465#
$UNITM   001010                378#
$USWR    001262                472#
$VECT1   001306                497#
$VECT2   001310                498#
$XTSTR   035162               5569#
$$GET4=  000000               5175#
$0FILL   037521               6033#   6037#   6047    6082#
$40CAT=  ****** U             5566    5638
.      = 056334                340#    344#    348     349#    356#    357#    363     364#    366#    368#    384     385#    387#
                               389#    399#    578     593     594    1350#   1989#   2811#   3121#   3445#   3615#   4783    4786
                              4791    5103    5107    5609    5610    5611    5663    5664    5665    5711#   5870#   5950    6002#
                              6087    6179#   6180    6186    6239    6296    6318
.$ASTA=  ****** U             5954    5957
.$X    = 001000                363#    368
```

```
MD-11-DOKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 166
DOKKAA,P11     07-FEB-77 11:01               CROSS REFERENCE TABLE -- MACRO NAMES

AJP       529#    1426    1623    1986    2170    2849    3118    3294    3442    3612    3952    4452    4911
COMMEN    139#
CSWP      532#    4717    4729
ENDCOM    139#
ERROR      33#     983    1075    1111    1117    1152    1167    1220    1240    1254    1278    1289    1300    1313    1338
         1344    1377    1387    1397    1477    1486    1523    1538    1581    1651    1692    1716    1770    1789    1800
         1806    1847    1870    1879    1897    1919    1930    1968    1975    2175    2181    2188    2194    2297    2308
         2316    2323    2334    2415    2446    2459    2467    2472    2645    2651    2658    2664    2767    2778    2786
         2793    2804    2934    2957    2965    2972    2983    3067    3078    3090    3103    3113    3221    3234    3242
         3247    3252    3379    3402    3410    3417    3428    3554    3567    3575    3588    3594    3608    3724    3737
         3745    3758    3764    3778    3843    3850    3856    3884    3894    3903    3911    4000    4029    4039    4048
         4056    4118    4135    4164    4172    4181    4191    4243    4249    4295    4299    4357    4361    4506    4515
         4544    4552    4561    4571    4623    4629    4635    4675    4741    4810    4856    4859    4999    5007    5009
         5127    5135    5138    5215    5219    5228    5235    5257
ESCAPE    139#
GET       530#    1675    1706    1764    1784    1795    1867    1876    1915    1926    1961    2287    2290    2313    2319
         2433    2456    2464    2757    2760    2783    2789    2926    2929    2942    2960    3059    3062    3074    3086
         3093    3200    3231    3239    3371    3374    3407    3413    3542    3545    3551    3564    3572    3712    3715
         3721    3734    3742    3877    3880    3900    3907    4022    4025    4045    4052    4157    4160    4169    4178
         4494    4537    4540    4549    4558    4844    4853    5201    5204    5212    5231
GETPRI    139#
GETSWP    139#
GNI       533#    1678    1709    1759    2282    2428    2752    2921    3054    3203    3366    3537    3707    3872    4017
         4152    4489    4532    4840    5196
GTA       534#    1693    1699    1769    2303    2440    2773    2952    3215    3397    3583    3753    3889    4034    4185
         4565    5272
JP        528#    1426    1623    1986    2170    2849    3118    3294    3442    3612    3952    4452    4911
MSG1      536#     882
MSG10     543#    1317
MSG11     544#    1401
MSG12     545#    1502
MSG13     546#    1599
MSG14     547#    1813
MSG15     548#    1995    2479
MSG16     549#    2200    2670
MSG17     550#    2340    3126
MSG2      537#    1091
MSG20     551#    2815    3260
MSG21     552#    3000
MSG22     553#    3451    3621
MSG23     554#    3919
MSG24     555#    4067
MSG25     556#    4207
MSG26     557#    4254
MSG27     558#    4304
MSG3      538#    1131
MSG30     559#    4366
MSG31     560#
MSG32     561#    4582
MSG33     562#    4648
MSG34     563#    4686
MSG35     564#    4749
MSG36     565#    4884
MSG37     566#    5014
MSG4      539#    1202
MSG40     567#
```

```
MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 167
DQKKAA.P11    07-FEB-77 11:01           CROSS REFERENCE TABLE -- MACRO NAMES

MSG42     569#    3784
MSG43     569#    1995
MSG5      540#    1355
MSG6      541#    1225
MSG7      542#    1258
MULT      139#
NEWTST    139#     890    1089    1129    1200    1223    1256    1315    1353    1399    1500    1597    1811    1993    2198
         2338     2477    2668    2813    2998    3124    3258    3449    3619    3782    3917    4065    4205    4252    4302
         4364     4580    4646    4684    4747    4882    5012
POP       139#    5857    5996    5997    6228    6305    6306
PUSH      139#    5816    5957    5959    5980    6202    6286    6292
REPORT    139#
SAV       525#    1150    1217    1276    1287    1296    1309
SCOPE      34#     897    1104    1143    1208    1234    1267    1325    1368    1422    1513    1619    1831    2027    2228
         2365     2508    2698    2845    3013    3151    3290    3476    3646    3800    3948    4089    4215    4263    4317
         4387     4591    4659    4698    4773    4907    5040    5158
SETPRI    139#
SETTRA   6262#    6271    6272    6273    6274    6277    6278    6279
SETUP     139#     572
SKIP      139#    1061    1194    1216    1245    1249    1279    1290    1306    1334    1339    1347    1388    1393    1479
         1488     1594    2418    2640    2808    3838    4061    4195    4238    4290    4320    4352    4672    4676    4726
         4743     4776    4829    4976    4980
SKT       535#     898    1106    1144    1209    1235    1268    1326    1369    1423    1514    1620    1832    2028    2229
         2366     2509    2699    2846    3014    3152    3291    3477    3647    3801    3949    4090    4216    4264    4318
         4388     4592    4660    4699    4774    4908
SLASH     139#     618     621     656     658     700     702     716     718     758     760     790     792    5188    5190
         5248     5250    5269    5271    5309    5311    5321    5324    5337    5340    5353    5356    5367    5370    5394
         5397     5406    5409    5454    5457    5503    5506    5542    5545
SPACE     139#
STARS      27     139#     337     347     360     362     369     382     457     569     880     895    1089    1102    1129
         1141     1200    1206    1223    1232    1256    1265    1315    1323    1353    1366    1399    1420    1500    1511
         1597     1617    1811    1829    1993    2025    2198    2226    2338    2363    2477    2506    2668    2696    2813
         2843     2998    3011    3124    3149    3258    3288    3449    3474    3619    3644    3782    3798    3917    3946
         4065     4087    4205    4213    4252    4261    4302    4315    4364    4385    4580    4589    4646    4657    4684
         4696     4747    4771    4863    4879    4882    4905    5012    5030    5151    5554    5613    5667    5806    5873
         5952     6009    6086    6092    6121    6188    6241    6282    6298    6323    7455    7931    7933
SWP       531#    1530    1541    1564    1573    1639    1656    1725    1742    1978    2161    2266    2632    2736    3810
         4617     4637
SWRSU     139#    595#
TPMTRP   6262#
TYPBIN    139#
TYPDEC    139#    5170
TYPNAM    139#
TYPNUM    139#
TYPOCS    139#
TYPOCT    139#    5679    5703
TYPTXT    139#
UNLY      526#     863    1070    1540    1583    1663    1749    1852    1902    1950    2060    2096    2149    2273    2419
         2536     2572    2620    2743    2900    3041    3194    3353    3524    3694    3812    3860    4005    4140    4223
         4273     4335    4497    4520    4604    4812    4832    5237    5258    5792
$$ESCA    139#
$$NEWT    139#     890    1089    1129    1200    1223    1256    1315    1353    1399    1500    1597    1811    1993    2198
         2338     2477    2668    2813    2998    3124    3258    3449    3619    3782    3917    4065    4205    4252    4302
         4364     4580    4646    4684    4747    4882    5012
$$SFT    6262#    6271    6272    6273    6274    6277    6278    6279
$$SFTM    611#


MD-11-DQKKA-A 11/6X CACHE  DIAGNOSTIC   MACY11 27(1006)  09-FEB-77  15:33  PAGE 168
DQKKAA.P11    07-FEB-77 11:01           CROSS REFERENCE TABLE -- MACRO NAMES

$$SKIP    139#    1061    1194    1216    1245    1249    1279    1290    1306    1347    1388    1393    1479    1488    1594
         2418     2640    2808    3838    4061    4195    4238    4290    4320    4352    4672    4676    4726    4743    4776
         4829     4976    4980
.EQUAT      2#      29
.HEADE      2#
.KT11       2#     139
.SETUP      2#     337
.SWPHI      2#      13
.SWPLO     25#
.$ACTI      2#     380
.$APTB    455
.$APTH      2#     350
.$APTY      2#    5950
.$CATC      2#     330
.$CMTA      2#
.$EOP       2#    5149
.$ERRO      2#    5611
.$ERRT      2#    5665
.$POWE      2#    6280
.$RDOC      2#    6196
.$READ      2#    6084
.$SCOP      2#    5552
.$TRAP      2#    6239
.$TYPD      2#    5804
.$TYPE      2#    5871
.$TYPO      2#    6007


. ABS.  056334      000


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DSK2:DQKKAA,DSK2:DQKKAA/SOL/CRF=DSK2:DQKKAA.P11
 RUN-TIME: 25 22 2 SECONDS
 RUN-TIME RATIO: 487/51=9.5
 CORE USED:  34K  (60 PAGES)
```