KDJ11-A KDJ11 CACHE MEM SYS
CZKOMBO

AH-T711B-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA

```
                                                        .REM &


                        IDENTIFICATION
                        --------------
            PRODUCT CODE:   AC-T710B-MC

            PRODUCT NAME:   CZKDMB0 KDJ11 CACHE MEMORY DIAGNOSTIC

            PRODUCT DATE:   15-MAR-84

            MAINTAINER:     DIAGNOSTIC ENGINEERING

            AUTHORS:        HENRY ENMAN, JIM PITTMAN, BARRY IRRGANG





                                                        &
```

```
41
42                                                          .REM &
43                                          HISTORY
44                                          -------
45
46          OCT-83  REV. A          FIRST RELEASE
47          FEB-84  REV. B          CORRECTIONS MADE TO:
48                                  1. CORRECT VECTOR AREA MAINTENANCE PROBLEM
49                                  2. ALLOW DIAGNOSTIC TO EXECUTE PROPERLY IN APT
50                                     ENVIRONMENT.
51                                  3. ALLOW DIAGNOSTIC TO BE EXECUTED IN SYSTEMS WITH
52                                     MXV11-A/B MODULES. REQUIRES DELETION OF TESTS
53                                     WHICH CAUSE KDJ11-A CACHE PARITY ABORTS.
54                                  4. ENSURE THAT CPU ERROR REGISTER IS CLEARED AFTER
55                                     COMPLETION OF TEST THAT MIGHT CAUSE IT TO BE SET.
56                                  5. SAVE PC AND CONTENTS OF R6 ON UNEXPECTED INTERRUPTS
                                                                                    &
```

.REM &

TABLE OF CONTENTS

&

```
                                                                .REM &
                1.0   GENERAL INFORMATION

                1.1   PROGRAM ABSTRACT

                THIS IS AN APT COMPATIBLE VERSION OF THE KDJ11 CACHE MEMORY
                SYSTEM DIAGNOSTIC. IT FOCUSES ON TESTING THE FUNCTIONALITY
                OF THE CACHE MEMORY SYSTEM. A SWITCH IS PROVIDED IN THE
                SOFTWARE SWITCH REGISTER TO DISABLE 22 BIT ADDRESS GENERATION
                IN 18 BIT QBUS SYSTEMS. THIS IS IMPLEMENTED BY SETTING BIT 08
                TO A ONE. DEFAULT IS TO TEST 22 BIT ADDRESSES. IN ADDITION;
                A SWITCH IS PROVIDED TO ENABLE THE EXECUTION OF THE CACHE DATA,
                AND CACHE TAG RAM DATA RELIABILITY TESTS. THESE TESTS ARE VERY
                LONG AND MAY NOT BE DESIRED IN ALL APPLICATIONS. THESE TESTS
                ARE ENABLED BY SETTING BIT 09 IN THE SOFTWARE SWITCH REGISTER
                (LOCATION 176) TO ONE. DEFAULT IS TO NOT EXECUTE THESE TESTS.

                1.2   SYSTEM REQUIREMENTS

                KDJ11-A PROCESSOR MODULE
                ENSURE THAT HALT TRAP OPTION IS DISABLED (JUMPER W9 INSTALLED)
                32KW MEMORY
                Q-22 BACKPLANE (18 BIT QBUS MAY BE USED WITH REDUCED TEST COVERAGE)
                SERIAL LINE UNIT AND CONSOLE TERMINAL (CONSOLE TERMINAL NOT REQUIRED FOR APT)

                1.3   RELATED DOCUMENTS AND STANDARDS

                KDJ11-A MODULE SPECIFICATION REV 2.2
                PDP11 MAINDEC SYSMAC PACKAGE
                J11 CONTROL CHIP SPECIFICATION 21-17679-00
                J11 DATA CHIP SPECIFICATION 21-17677-00

                1.4   DIAGNOSTIC HIERARCY PREREQUISITES

                THE KDJ11 CPU, MEMORY MANAGEMENT AND FLOATING POINT DIAGNOSTICS
                SHOULD RUN SUCCESSFULLY PRIOR TO RUNNING THE CACHE MEMORY
                SYSTEM TESTS.

                1.5   ASSUMPTIONS

                IT IS ASSUMED THAT THE DIAGNOSTIC OPERATOR IS FAMILIAR WITH
                THE XXDP+ OPERATING SYSTEM AND THE J11 MICRO-ODT.

                2.0   OPERATING INSTRUCTIONS

                2.1   LOADING AND STARTING PROCEEDURE

                LOAD PROGRAM INTO MEMORY USING STANDARD XXDP+ PROCEEDURES.
                THE PROGRAM IS STARTED BY LOADING ADDRESS 200 AND USING
                THE J11 MICRO-ODT G COMMAND TO START. THE PROGRAM
                IDENTIFICATION MESSAGE WILL BE TYPED AFTER THE FIRST PASS
                OF THE COMPLETE PROGRAM.

                2.2   PROGRAM OPTIONS
```

THE FOLLOWING ASSIGNMENTS HAVE BEEN MADE FOR THE KDJ11-A
DIAGNOSTIC SWITCH REGISTER BITS:

BIT#15	14	13	12	11	10	9	8
					DON'T TEST BEVENT	18 BIT ADDRESS ONLY	EXTENDED CACHE TESTS

DEFAULT SETTINGS ARE TO TEST 22 BIT ADDRESSES AND NOT DO THE
EXTENDED CACHE TESTS.

PRIOR TO EXECUTING THE FIRST PASS OF THE DIAGNOSTIC THE OPERATOR
WILL BE DIRECTED TO SET THE SWITCH REGISTER TO INDICATE WHETHER
THE KDJ11-A UNDER TEST IS IN A SYSTEM CONFIGURED FOR 18 OR 22 BIT
ADDRESSING AND WHETHER CACHE RAM DATA RELIABILITY TESTS ARE TO BE
EXECUTED. AN 18 BIT ADDRESS CONFIGURATION SHOULD BE INDICATED IF
ANY 18 ADDRESS BIT ONLY MEMORY BOARDS RESIDE IN THE SYSTEM OR IF
THE SYSTEM BACKPLANE DOES NOT SUPPORT 22 ADDRESS BITS. THE CACHE
RAM DATA RELIABILTY TESTS REQUIRE APPROXIMATELY 35 MINUTES TO
EXECUTE AND SHOULD ONLY BE SELECTED IF PROBLEMS WITH CACHE DATA
CORRUPTION OR RETENTION ARE SUSPECTED.

TO CHANGE THE SWITCH REGISTER; HALT THE PROGRAM AND RESTART IT
AT 200 ANSWERING THE INITIAL QUESTIONS.

2.3 OPERATION UNDER APT

OPERATION IN THE APT ENVIRONMENT REQUIRES SOME SPECIAL CONSIDERA-
TIONS DUE TO THE ASYNCHRONOUS HALTS OF THE DIAGNOSTIC BY THE APT
MONITOR. IF THE EFFECTS OF THESE HALTS ARE NOT ANTICIPATED, FALSE
ERRORS MAY BE REPORTED. THEREFORE, WHEN OPERATING IN THE APT ENVIRON-
MENT THE FOLLOWING DIFFERENCES IN THE EXECUTION OF THE PROGRAM SHOULD
BE NOTED:

1. CERTAIN CACHE TESTS WHICH RELY ON THE INTEGRITY OF DATA IN THE
   HIT/MISS REGISTER OR THE WAY IN WHICH THE CACHE IS ALLOCATED WILL
   JUMP TO A SUB-ROUTINE IF AN ERROR IS ENCOUNTERED. THE SUBROUTINE
   DETERMINES IF THE DIAGNOSTIC IS OPERATING IN APT ENVIRONMENT. IF
   IT IS IN APT ENVIRONMENT IT WILL RETRY THE FAILING TEST ONE TIME.
   IF THE TEST PASSES ON THE SECOND ATTEMPT THE FIRST ERROR WILL BE
   CONSIDERED TO BE APT INDUCED. IF IT FAILS ON THE SECOND ATTEMPT
   THE ERROR ROUTINE WILL BE CALLED.

2.4 EXECUTION TIMES

A. THE LONGEST TEST

THE LONGEST TEST IS THE TAG RAM DATA RELIABILITY TEST WHICH TAKES
APPROXIMAELY 25 MINUTES TO EXECUTE. FOR THIS REASON THE DATA RELIABILITY
TESTS ARE NORMALLY DE-SELECTED.

B. FULL PASS TIME

THE TIME FOR A FULL PASS WITH THE DATA RELIABILITY TESTS DE-SELECTED

```
196                 IS APPROXIMATELY 5 SECONDS. WITH ALL TESTS SELECTED A FULL PASS
197                 TAKES ABOUT 35 MINUTES.
198
199                 3.0   ERROR INFORMATION
200
201                 ERRORS WILL CAUSE THE FOLLOWING ERROR MESSAGE TO BE PRINTED:
202
203                             CACHE SYSTEM ERROR
204                             ERROR # = (UNIQUE ERROR NUMBER)
205                             ERROR PC = (PC AT TIME OF ERROR)
206
207                 THE ERROR WILL THEN BE REPORTED TO APT AND THE PROGRAM
208                 WILL HALT.
209
210                 4.0   PROGRESS REPORT
211
212                 AT THE END OF EACH PASS THE DIAGNOSTIC NAME AND PASS COUNT ARE PRINTED.
213
214                                                       ε
215
```

```
                                .TITLE PROGRAM HEADER AND TABLES
                                .SBTTL  PROGRAM HEADER

                                        .MCALL  ETXT1,ETXT2,ETXT3,ETXT4,ETXT5,ETXT6,ETXT7,ETXT8,ETXT9
                                        .MCALL  ETXT10,ETXT11,ETXT12,ETXT13,ETXT14,ETXT15,ETXT16,ETXT17
                                        .MCALL  ETXT18,ETXT19,ETXT20,ETXT21,ETXT22,ETXT23,ETXT24,ETXT25
                                        .MCALL  ETXT26,ETXT27,ETXT28,ETXT29,ETXT30,ETXT31,ETXT32,ETXT33
                                        .MCALL  ETXT34,ETXT35,ETXT36,ETXT37,ETXT38,ETXT39,ETXT40,ETXT41
                                        .MCALL  ETXT42,ETXT43,ETXT44,ETXT45,ETXT46,ETXT47,ETXT48,ETXT49
                                        .MCALL  ETXT50,ETXT51,ETXT52,ETXT53,ETXT54,ETXT55,ETXT56,ETXT57
                                        .MCALL  ETXT58,ETXT59,ETXT60,ETXT61,ETXT62
                                        .MCALL  ERRF1,ERRF2,ERRF3,ERRF4,ERRF5,ERRF6,ERRF7
                                        .MCALL  ERRF8,ERRF9,ERRF10,ERRF11,ERRF12,ERRF13,ERRF14,ERRF15
                                        .MCALL  ERRF16,ERRF17,ERRF18,ERRF19,ERRF20,ERRF21,ERRF22,ERRF23
                                        .MCALL  ERRF24,ERRF25,ERRF26,ERRF27,ERRF28,ERRF29,ERRF30,ERRF31
                                        .MCALL  ERRF32,ERRF33,ERRF34,ERRF35,ERRF36,ERRF37,ERRF38,ERRF39
                                        .MCALL  ERRF40,ERRF41,ERRF42,ERRF43,ERRF44,ERRF45,ERRF46,ERRF47
                                        .MCALL  ERRF48,ERRF49,ERRF50,ERRF51,ERRF52,ERRF53,ERRF54,ERRF55
                                        .MCALL  ERRF56,ERRF57,ERRF58,ERRF59,ERRF60,ERRF61,ERRF62,ERRF63
                                        .MCALL  ERRF64,ERRF65,ERRF66,ERRF67,ERRF01,ERRF02,ERRF03,ERRF04
                                        .MCALL  ERRF05,ERRF06,ERRF07,ERRF08,ERRF09,SFTERR,EXITST
                                        .MCALL  NEWTST,ERRDEF,,EQUAT,,KT11,,$40CAT,,$EOP,,$APTBLS,SETUP
                                        .MCALL  .$TYPE,,$TYPDEC,ERRDF,BGNTST,ENDTST,BGNMOD,ENDMOD,CKLOOP
                                        .MCALL  .HEADER,,SETUP,,$TRAP,BGNSUB,ENDSUB,,$ACT11,,$APTHDR
                                        .MCALL  .$APTYPE,,$ERROR,,$TYPOCT,,$READ



                                .TITLE  KDJ11-A CACHE MEMORY SYSTEM TEST
                                ;*COPYRIGHT (C) MARCH,1984
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
```

```
                        .TITLE GLOBAL AREAS
                        .SBTTL  GLOBAL EQUATES SECTION
                        ;++
                        ; THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
                        ; ARE USED IN MORE THAN ONE TEST.
                        ;--
                        .SBTTL  BASIC DEFINITIONS
                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
001000                  STACK=  1000
                        .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
17757C                  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
```

```
                        EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        TRAPVEC=34              ;;"TRAP" TRAP
                        TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        TPVEC=  64              ;;TTY PRINTER VECTOR
                        PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                        ;*KT11 VECTOR ADDRESS

                        MMVEC=  250

                        ;*KT11 STATUS REGISTER ADDRESSES

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

                        ;*USER "I" PAGE DESCRIPTOR REGISTERS

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

                        ;*USER "D" PAGE DESCRIPTOR REGISTORS

                        UDPDR0= 177620
                        UDPDR1= 177622
                        UDPDR2= 177624
                        UDPDR3= 177626
                        UDPDR4= 177630
                        UDPDR5= 177632
                        UDPDR6= 177634
                        UDPDR7= 177636

                        ;*USER "I" PAGE ADDRESS REGISTERS

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

                        ;*USER "D" PAGE ADDRESS REGISTERS

                        UDPAR0= 177660
                        UDPAR1= 177662
                        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

                                        ;THESE ARE FLOATING POINT ACCUMULATOR EQUATES
                000000                  AC0=    %0
                000001                  AC1=    %1
                000002                  AC2=    %2
                000003                  AC3=    %3
                000004                  AC4=    %4
                000005                  AC5=    %5
                000006                  AC6=    %6
                000007                  AC7=    %7

                000244                  FPVEC=  244

                                        ;THESE ARE CACHE REGISTER EQUATES
                177746                  CCR=    177746          ;CACHE CONTROL REGISTER
                177744                  MSER=   177744          ;MEMORY SYSTEM ERROR REGISTER
                177752                  HITMIS= 177752          ;HIT/MISS REGISTER
                177766                  CPEREG= 177766          ;CPU ERROR REGISTER
```

```
                        ;MISCELANEOUS DEFINITIONS
177546                  BEVENT= 177546          ;BEVENT CONTROL REGISTER
177560                  RCSR=   177560
177562                  RBUF=   177562
177564                  XCSR=   177564
177566                  XBUF=   177566
000000                  ERRTN=  HALT
000001                  $TSTNU=1
000001                  ERRNUM= 1               ;INITIALIZE ERROR NUMBER COUNTER
002000                  ASWREG= 2000            ;SWR FOR APT--NO BEVENT TESTING


                        ;THIS EQUATE DEFINES THE BOTTOM OF THE PROGRAM STACK POINTER
001000                          STBOT=  1000
000000                  .ASECT
                        .SBTTL  TRAP CATCHER

000000                          .=0
                        ;*ALL UNUSED LOCATIONS OF THE VECTOR AREA CONTAIN
                        ;*A ".+2, IOT" SEQUENCE TO CATCH AND PROCESS ILLEGAL
                        ;*TRAPS AND INTERRUPTS THAT MIGHT OCCUR.
                        ;*THE IOT TRAP WHICH IS TAKEN ON THE ILLEGAL TRAP/INT
                        ;*TRAPS TO THE $SCOPE ROUTINE WHICH (IF THE RETURN PC IS
                        ;*LESS THAN 1002) JUMPS TO THE $ERROR ROUTINE.
                        ;*THE $ERROR ROUTINE WILL REPORT THE ERROR AS FOLLOWS:
                        ;*      PC=YYYYYY UNEXPECTED TRAP TO XXX
                        ;*AND RETURN TO THE PROGRAM AT PC=YYYYYY+2
                        ;*WHERE XXX=LOCATION OF ILLEGAL TRAP
                        ;*      YYYYYY=PC AT TIME OF TRAP
                        ;*NOTE: IF THE PROCESSOR IS NOT AN 11/05 THE PROGRAM
                        ;*      CAN BE STARTED AT ADDRESS 0 AS WELL AS ADDRESS 200.
000000  000000          $4OCAT: HALT                    ;;HALT
000002  000737                  BR      .-100           ;;BRANCH TO 177700 & TIME OUT (NOT ON
                                                        ;;11/05)
000004  001716                  .WORD   START           ;;VECTOR TO STARTING ADDRESS
000006  000340                  .WORD   340             ;;WITH PRIORITY LEVEL 7
        000174                  .=174
000174  000000          DISPREG:.WORD   0               ;;SOFTWARE DISPLAY REGISTER
000176  000000          SWREG:  .WORD   0               ;;SOFTWARE SWITCH REGISTER
                        .SBTTL  STARTING ADDRES(ES)
000200  000137  001716          JMP     @#START ;;GO TO START OF PROGRAM
                        .SBTTL  ACT11 HOOKS

                        ;;**************************************************************
                        ;HOOKS REQUIRED BY ACT11
        000204                  $SVPC=.                 ;SAVE PC
        000046                  .=46
000046  011034                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                  .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
        000204                  .=$SVPC                 ;; RESTORE PC
                        .SBTTL  APT PARAMETER BLOCK

                        ;;**************************************************************
                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
```

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

000204                          $APTHD:
000204  000000                $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
000206  001000                $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
000210  000001                $TSTM:  .WORD   1       ;;RUN TIM OF LONGEST TEST
000212  000004                $PASTM: .WORD   4       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
000214  000000                $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
000216  000014                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
000204                          .=.$X                   ;SAVE CURRENT LOCATION COUNT
000002                          .=2
000002  000000                0
000004  000006                6
000006  000004                4                               ;SET UP SOME VECTORS
000204                          .=.$X                   ;RESTORE LOCATION COUNT
001000                          .=1000
```

```
                        .SBTTL  GLOBAL DATA SECTION

                        ;++
                        ; THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
                        ; IN MORE THAN ONE TEST.
                        ;--
                        .SBTTL  APT MAILBOX-ETABLE

                        ;;*****************************************************************
                        .EVEN
001000                  $MAIL:                          ;;APT MAILBOX
001000  000000          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001002  000000          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001004  000000          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001006  000000          $PASS:  .WORD   APASS           ;;PASS COUNT
001010  000000          $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001012  000000          $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001014  000000          $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001016  000000          $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
001020                  $ETABLE:                        ;;APT ENVIRONMENT TABLE
001020     000          $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
001021     000          $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
001022  002000          $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
001024  000000          $USWR:  .WORD   AUSWR           ;;USER SWITCHES
001026  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
001030                  $ETEND:
                        .MEXIT

                        ;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE VECTOR DATA
                        ;WHEN THE TEST NEEDS TO HAVE AN ERROR CONDITION RESPOND DIFFERENTLY
                        ;FROM THE DEFAULT RESPONCE.
001030  000000          SLOC00: .WORD   0
001032  000000          SLOC01: .WORD   0

                        ;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE WORKING DATA.
001034  000000          EXPDAT: .WORD   0               ;STORES EXPECTED (GOOD) DATA FOR COMPARISONS
001036  000000          RECDAT: .WORD   0               ;STORES RECIEVED DATA TO BE VERIFIED
001040  000000          COUNT:  .WORD   0               ;ERROR INDICATOR FOR FLOATING POINT TESTS
001042  000000          FLAG:   .WORD   0               ;USED TO STORE "FLAG" CONDITIONS
001044  000000          ERRCNT: .WORD   0               ;STORAGE FOR ERROR COUNT
001046  177570          SWR:    .WORD   DSWR            ;STORAGE FOR SWITCH REGISTER ADDRESS
001050  177570          DISPLAY:.WORD   DDISP           ;STORAGE FOR DISPLAY REGISTER ADDRESS
001052  000000          $ERFLG: .WORD   0               ;ERROR FLAG
                        ;THESE LOCATIONS ARE USED BY MORE THAN ONE TEST AS LOOP COUNTERS
001054  000000          DCOUNT: .WORD   0
001056  000000          ALLCTR: .WORD   0
001060  000000          LOOPIN: .WORD   0
001062  000000          SAVSP1: .WORD   0               ;STORAGE FOR UNEXPECTED TRAP DATA
001064  000000          SAVSP2: .WORD   0               ;       "       "       "       "
```

```
672  001066  000000          FWDSEQ: .WORD   0       ;USED TO INDICATE DIRECTION OF ADDRESSING
673  001070  000000          ADDLSB: .WORD   0       ;STORES LEAST SIGNIFICANT BIT FOR RAM TESTS
674  001072  000000          RITEDA: .WORD   0       ;STORES WRITE DATA FOR RAM TESTS
675  001074  000000          NEWDAT: .WORD   0       ;DATA STORE FOR RAM TESTS
676  001076  000000          CURDAT: .WORD   0       ;DATA STORE FOR RAM TESTS
677  001100  000000          FSTADD: .WORD   0       ;STORES FIRST ADDRESS IN ADDRESSING SEQUENCE
678  001102  000000          LSTADD: .WORD   0       ;STORES LAST ADDRESS IN ADDRESSING SEQUENCE
679  001104  000000          CURADD: .WORD   0       ;STORES CURRENT ADDRESS FOR RAM TESTS
680  001106  000000          LOWADD: .WORD   0       ;STORES LOW ADDRESS FOR RAM TESTS
681  001110  000000          GOODAD: .WORD   0       ;STORES GOOD ADDRESS FOR RAM TESTS
682  001112  000000          TSTADD: .WORD   0       ;ADDRESS STORE FOR RAM TESTS
683  001114  000000          NEWADD: .WORD   0       ;ADDRESS STORE FOR RAM TEST
684  001116  000000          SOFTER: .WORD   0       ;USED TO STORE SOFT ERROR COUNT
685  001120  000000          SOFTRE: .WORD   0       ;USED TO HOLD RETRY ADDRESS ON SOFT ERRORS
686
687
688
689
690
691
692
693
694                          ;!!!!!THIS IS IT. THE PROGRAM TEST LOCATION AND WRITE BUFFER!!!!!!!!!!!!!!!!!!
695  001122                  TSTLOC:
696  001122  000020                  .BLKW   20
```

```
                                        .SBTTL  GLOBAL TEXT SECTION

                                        ;++
                                        ; THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
                                        ; MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
                                        ; MORE THAN ONE TEST.
                                        ;--

                                        ;
                                        ; FORMAT STATEMENTS USED IN PRINT CALLS
                                        ;

001162  005015  042523  020124  OPMSG2: .ASCIZ  <CR><LF>/SET BIT 8 = 1 FOR 18 BIT SYSTEM/
001170  044502  020124  020070
001176  020075  020061  047506
001204  020122  034061  041040
001212  052111  051440  051531
001220  042524  000115
001224  005015  042523  020124  OPMSG3: .ASCIZ  <CR><LF>/SET BIT 9 = 1 FOR CACHE RAM AND TAG RAM DATA RELIABILITY TESTS/
001232  044502  020124  020071
001240  020075  020061  047506
001246  020122  040503  044103
001254  020105  040522  020115
001262  047101  020104  040524
001270  020107  040522  020115
001276  040504  040524  051040
001304  046105  040511  044502
001312  044514  054524  052040
001320  051505  051524     000
001325     015  041412  041501  ERRMSG::.ASCIZ  <CR><LF>/CACHE SYSTEM ERROR/
001332  042510  051440  051531
001340  042524  020115  051105
001346  047522  000122
001352  005015  051105  047522  ERR1:   .ASCIZ  <CR><LF>/ERROR @ =/
001360  020122  020043  000075
001366  005015  051105  047522  ERR2:   .ASCIZ  <CR><LF>/ERROR PC =/
001374  020122  041520  036440
001402     000
001403     015  020012  020040  $CRLF:  .ASCIZ  <CR><LF>/   /
001410     000
001412                                  .EVEN
```

```
                        .SBTTL  GLOBAL ERROR REPORT SECTION

                        ;++
                        ; THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
                        ; USED BY MORE THAN TEST TO OUTPUT ADDITIONAL ERROR INFORMATION.
                        ;--
```

```
                        .SBTTL  GLOBAL SUBROUTINES SECTION

                        ;++
                        ; THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
                        ; THAT ARE USED IN MORE THAN ONE TEST.
                        ;--
                        ;++
                        ; FUNCTIONAL DESCRIPTION:
                        ;    SUBROUTINE TO INITIALIZE ALL THE MMU REGISTERS


                        ; INPUTS: NONE

                        ; OUTPUTS: NONE

                        ; SUBORDINATE ROUTINES USED: LOAD PARS
                        ;                            LOAD PDRS

                        ; FUNCTIONAL SIDE EFFECTS: NONE

                        ; CALLING SEQUENCE:     JSR     PC,INITMM

001412  012701  172240  INITMM: MOV     #172240,R1              ;BASE ADDRESS OF SIPARS
001416  004767  000132          JSR     PC,     LDPARS
001422  012701  172260          MOV     #172260,R1              ;BASE ADDRESS OF SDPARS
001426  004767  000122          JSR     PC,     LDPARS
001432  012701  172340          MOV     #172340,R1              ;BASE ADDRESS OF KIPARS
001436  004767  000112          JSR     PC,     LDPARS
001442  012701  172360          MOV     #172360,R1              ;BASE ADDRESS OF KDPARS
001446  004767  000102          JSR     PC,     LDPARS
001452  012701  177640          MOV     #177640,R1              ;BASE ADDRESS OF UIPARS
001456  004767  000072          JSR     PC,     LDPARS
001462  012701  177660          MOV     #177660,R1              ;BASE ADDRESS OF UDPARS
001466  004767  000062          JSR     PC,     LDPARS
001472  012701  177600          MOV     #177600,R1              ;BASE ADDRESS OF UIPDRS
001476  004767  000102          JSR     PC,     LDPDRS
001502  012701  177620          MOV     #177620,R1              ;BASE ADDRESS OF UDPDRS
001506  004767  000072          JSR     PC,     LDPDRS
001512  012701  172300          MOV     #172300,R1              ;BASE ADDRESS OF KIPDRS
001516  004767  000062          JSR     PC,     LDPDRS
001522  012701  172320          MOV     #172320,R1              ;BASE ADDRESS OF KDPDRS
001526  004767  000052          JSR     PC,     LDPDRS
001532  012701  172200          MOV     #172200,R1              ;BASE ADDRESS OF SIPDRS
001536  004767  000042          JSR     PC,     LDPDRS
001542  012701  172220          MOV     #172220,R1              ;BASE ADDRESS OF SDPDRS
001546  004767  000032          JSR     PC,     LDPDRS
001552  000207                  RTS     PC                      ;RETURN
```

```
                                        ;++
                                        ; FUNCTIONAL DESCRIPTION:
                                        ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE ADDRESS REGISTERS (PARS).
                                        ;       THIS ROUTINE WILL INITIALIZE 8 PARS STARTING AT A BASE ADDRESS
                                        ;       SUPPLIED BY THE CALLING ROUTINE. PARS 0-5 WILL BE MAPPED FROM
                                        ;       ADDRESS 0 TO ADDRESS 137777 (0-24K). PAR 6 WILL BE MAPPED FROM
                                        ;       ADDRESS 200000 TO 217777 AND PAR 7 WILL BE MAPPED TO THE I/O
                                        ;       PAGE.
                                        ;
                                        ; INPUTS:
                                        ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PARS TO BE INITIALIZED
                                        ;
                                        ; OUTPUTS: NONE
                                        ;
                                        ; SUBORDINATE ROUTINES USED: NONE
                                        ;
                                        ; FUNCTIONAL SIDE EFFECTS: NONE
                                        ;
                                        ; CALLING SEQUENCE:     JSR     PC,LDPARS
                                        ;
001554  012702  000006          LDPARS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
001560  005003                          CLR     R3                      ;INITIALIZE INDEX VALUE
001562  010321                  1$:     MOV     R3,     (R1)+           ;LOAD PARS
001564  062703  000200                  ADD     #200,   R3              ;INDEX IN 4K INCREMENTS
001570  077204                          SOB     R2,     1$              ;LOAD FIRST SIX PARS
001572  012721  002000                  MOV     #2000,  (R1)+           ;LET PAR6 MAP TO 200000
001576  012711  177600                  MOV     #177600,(R1)            ;LET PAR7 MAP TO I/O PAGE
001602  000207                          RTS     PC                      ;RETURN
```

```
                        ;++
                        ; FUNCTIONAL DESCRIPTION:
                        ;    SUBROUTINE TO INITIALIZE ALL THE MMU PAGE DECRIPTOR REGISTERS (PDRS).
                        ;       THIS ROUTINE WILL INITIALIZE 8 PDRS STARTING AT A BASE ADDRESS
                        ;       SUPPLIED BY THE CALLING ROUTINE. PDRS 0-5 WILL BE INITIALIZED TO
                        ;       4K READ/WRITE BYPASS AND PDRS 6 AND 7 WILL BE INITIALIZED TO
                        ;       4K READ/WRITE NO BYPASS.
                        ;       NOTE:   THERE IS NO NEED TO BYPASS ON I/O PAGE REFERENCES BECAUSE
                        ;               THE CACHE DOES NOT ALLOCATE ANY OF THESE REFERENCES.
                        ;
                        ; INPUTS:
                        ;       R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PDRS TO BE INITIALIZED
                        ;
                        ; OUTPUTS: NONE
                        ;
                        ; SUBORDINATE ROUTINES USED: NONE
                        ;
                        ; FUNCTIONAL SIDE EFFECTS: NONE
                        ;
                        ; CALLING SEQUENCE:     JSR     PC,LDPARS
                        ;
001604  012702  000006  LDPDRS: MOV     #6,     R2              ;LET LOOP COUNTER COUNT FIRST 6 PARS
001610  012721  177406  1$:     MOV     #177406,(R1)+           ;LOAD PDRS WITH 4K READ/WRITE BYPASS
001614  077203                  SOB     R2,     1$              ;LOAD FIRST SIX PDRS
001616  012721  077406          MOV     #77406, (R1)+           ;LET PAR6 BE 4K READ/WRITE NO BYPASS
001622  012711  077406          MOV     #77406, (R1)            ;LET PAR7 BE 4K READ/WRITE NO BYPASS ALSO
001626  000207                  RTS     PC                      ;RETURN
```

```
                        ;**
                        ; FUNCTIONAL DESCRIPTION:
                        ;    SUBROUTINE TO HANDLE PARITY ERROR ABORTS FROM THE RAM STORE RAM TESTS.
                        ;
                        ; INPUTS:
                        ;       MEMORY SYSTEM ERROR REGISTER CONTAINS BITS INDICATING FAILURE
                        ;
                        ; OUTPUTS: NONE
                        ;
                        ; SUBORDINATE ROUTINES USED: NONE
                        ;
                        ; FUNCTIONAL SIDE EFFECTS: NONE
                        ;
                        ; CALLING SEQUENCE: CALLED BY PARITY ABORT
                        ;       MOV     @#114,  SLOC00  ;SAVE CONTENTS OF PARITY ABORT VECTOR
                        ;       MOV     #DSPAR, @#114   ;LET VECTOR POINT TO PARITY ABORT ROUTINE
                        ;
                        ;       (CACHE PARITY ERROR OCCURS)
                        ;
001630                  RAMPAR:
001630  104000                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
001632  000001                  .WORD   1                       ;UNIQUE ERROR NUMBER
001634  001325                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                        ;CACHE SYSTEM ERROR
001636  000002                  RTI                             ;RETURN
```

```
                        ;**
                        ;  FUNCTIONAL DESCRIPTION
                        ;       SUBROUTINE TO HANDLE SOFT ERRORS THAT MAY BE CAUSED BY
                        ;       APT SYSTEM MONITORING DURING THE CACHE TEST. FOR EXAMPLE:
                        ;       IF A TEST IS USING THE HIT/MISS REGISTER TO MONITOR CACHE
                        ;       ACTIVITY, AND APT BREAKS IN, THE HIT/MISS REGISTER WILL BE
                        ;       MODIFIED. WHEN THE TEST RESUMES, A FALSE ERROR MAY BE CAUSED BY
                        ;       THE CORRUPTED HIT/MISS REGISTER. LIKEWISE, A TEST MAY INITIALIZE
                        ;       CACHE TO A KNOWN STATE (ALLOCATED/NOT ALLOCATED); AN APT
                        ;       BREAK MAY CAUSE CACHE STATE TO ALTER AND THE TEST TO FAIL.
                        ;
                        ;INPUTS
                        ;       SOFTRE CONTAINS RETURN ADDRESS TO BEGINNING OF TEST +4 TO
                        ;       TEST AGAIN ON ERROR

                        ;OUTPUTS: NONE

                        ;SUBORDINATE ROUTINES USED: NONE

                        ;FUNCTIONAL SIDE EFFECTS: NONE

                        ;CALLING SEQUENCE:      JSR     PC,APTSFT

001640  122767  000001  177152  APTSFT: CMPB    #APTENV,$ENV            ;ARE WE IN APT MODE?
001646  001022                          BNE     1$                      ;NO SOFT ERRORS IF NOT IN APT MODE
001650  005767  177242                  TST     SOFTER                  ;HAS A SOFT ERROR ALREADY OCCURRED?
001654  001017                          BNE     1$                      ;IF YES GO TO ERROR
001656  062706  000002                  ADD     #2,R6                   ;CLEAN UP THE STACK TO RETRY TEST
001662  005267  177230                  INC     SOFTER                  ;INCREMENT THE SOFT ERROR COUNTER
001666  005037  177744                  CLR     @#MSER                  ;CLEAR THE MSER
001672  005037  177572                  CLR     @#SR0                   ;TURN OFF MMU
001676  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
001702  012737  000400  177746          MOV     #BIT08,@#CCR            ;FLUSH THE CACHE, CLEAR THE CCR
001710  000177  177204                  JMP     @SOFTRE                 ;RETRY TEST
001714  000207                  1$:     RTS     PC                      ;GO TO ERROR. ERROR OCCURRED MORE
                                                                        ;THAN ONCE OR NOT IN APT MODE.
```

```
                                START:
001716                          
001716  012737  000014  177746          MOV     #14,@#CCR               ;SET CACHE TO FORCE MISS
                                .SBTTL  INITIALIZE THE COMMON TAGS
001724  012706  001000                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001730  012737  013314  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001736  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001744  012737  012766  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001752  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001760  005067  177022                  CLR     $PASS           ;;CLEAR THE PASS COUNT
001764  016767  007012  007002          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001772  105067  177054                  CLRB    $ERFLG          ;;CLEAR THE ERROR FLAG
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001776  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002002  012737  002036  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002010  012767  177570  177030          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002016  012767  177570  177024          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002024  022777  177777  177014          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002032  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002034  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002036  012716  002044          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002042  000002                          RTI
002044  012767  000176  176774  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002052  012767  000174  176770          MOV     #DISPREG,DISPLAY
002060  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

                                .MACRO  $$SETMAIL       ?$ARG1
                                        CLR     $PASS           ;;CLEAR PASS COUNT
                                        BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
                                        BEQ     $ARG1           ;;YES,USE NON-APT SWITCH
                                        MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
                                $ARG1:
                                .ENDM   $$SETMAIL
002064  005067  176716                  CLR     $PASS           ;;CLEAR PASS COUNT
002070  132767  000200  176723          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002076  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002100  012767  001022  176740          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002106                          67$:
002106  012737  013314  000020          MOV     #$ERROR,@#IOTVEC        ;;SET UP IOT VECTORS
002114  012737  000340  000022          MOV     #340,@#IOTVEC+2         ;;TO GO TO ERROR ROUTINE
002122  005037  177766                  CLR     @#177766                ;CLEAR CPU ERROR REGISTER
002126  104401  001162           TYPE   ,OPMSG2                 ;OPERATOR MESSAGE 2
002132  104401  001224                  TYPE    ,OPMSG3         ;OPERATOR MESSAGE 3
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
002136  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
002142  001012                          BNE     68$             ;;BRANCH IF YES
002144  126727  176650  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
002152  001406                          BEQ     68$             ;;BRANCH IF YES
002154  026727  176666  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
002162  001005                          BNE     69$             ;;BRANCH IF NO
002164  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
002166  000403                          BR      69$
002170  112767  000001  010566  68$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
002176                          69$:
```

```
                                RESTART: CLR     $TESTN                  ;RESET $TESTN TO ZERO
                                         MOV     #14,@#CCR               ;SET CACHE TO FORCE MISS

                                ;;*************************************************************************
                                ;*TEST 1        CACHE REGISTER ACCESS TEST
                                ;;*************************************************************************
                                ;CHECK REGISTER ACCESS - THIS TEST WILL VERIFY EACH OF THE THREE
                                ;CACHE MEMORY SYSTEM REGISTERS CAN BE ACCESSED WITHOUT A TRAP TO
                                ;LOCATION 4(NON-EXISTANT ADDRESS TRAP) OCCURRING. THE REGISTERS TO
                                ;BE TESTED ARE:
                                ;       CACHE CONTROL           17777746
                                ;       MEMORY SYSTEM ERROR     17777744
                                ;       HIT/MISS                17777752
                                ;EACH REGISTER WILL BE ACCESSED WITH A WRITE CYCLE.
                                ;
                                ;BGNTST
                                ;SAVE CONTENT OF LOCATION 4
                                ;LET LOCATION 4 POINT TO ERROR ROUTINE
                                ;TEST THE CACHE REGISTERS
                                ;RESTORE CONTENTS OF LOCATION 4
                                ;EXIT   TST
                                ;
                                ;CACHE REGISTER ADDRESSES
                                ;       CCR=    17777746
                                ;       MSER=   17777744
                                ;       HITMIS= 17777752
                                ;ERROR ROUTINE: ERROR IN CACHE SYSTEM
                                ;               RETURN
                                ;ENDTST
                                ;;*************************************************************************
002210                          TST1:
002210  005267  176570                  INC     $TESTN                  ;INCREMENT TEST NUMBER
002214  013767  000114  176606          MOV     @#114,SLOCOO            ;SAVE PARITY INTERRUPT VECTORS
002222  012737  001630  000114          MOV     #RAMPAR,@#114           ;SET UP TRAP CATCHER FOR PARITY ERRORS.
002230  013704  000004                  MOV     @#4,R4                  ;SAVE CONTENTS OF VECTOR 4
002234  012737  002302  000004          MOV     #ERROUT,@#4             ;LET VECTOR 4 POINT TO ERROR ROUTINE
002242  005037  177746                  CLR     @#CCR                   ;TEST ACCESS TO CACHE CONTROL REG
002246  005037  177744                  CLR     @#MSER                  ;TEST ACCESS TO MEMORY SYSTEM ERR REG
002252  005037  177752                  CLR     @#HITMIS                ;TEST ACCESS TO HIT MISS REGISTER
002256  010437  000004                  MOV     R4,@#4                  ;RESTORE VECTOR
002262  005037  177744                  CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
002266  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
002272  005037  001116                  CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
002276  000167  000010                  JMP     TST2                    ;JUMP OVER DATA TABLES OR SUBROUTINES
                                                                        ;TO NEXT TEST

002302                          ERROUT:
002302  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
002304  000002                          .WORD   2                       ;UNIQUE ERROR NUMBER
002306  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                ;CACHE SYSTEM ERROR
002310  000002                          RTI                             ;CONTINUE FROM ERROR
```

```
                        ;CTST2:
                        ;;*****************************************************************
                        ;*TEST 2         TEST CACHE CNTRL REG BITS
                        ;;*****************************************************************
                        ;CCR REGISTER BIT TEST - THIS TEST WILL VERIFY THAT EACH READ/WRITE BIT OF
                        ;THE CACHE CONTROL REGISTER CAN BE SET AND CLEARED INDIVIDUALLY AND THAT
                        ;BITS 15 - 11 AND BIT 8 ARE ALWAYS READ AS ZEROS.
                        ;
                        ;BGNTST
                        ;INITIALIZE GOOD DATA TO #1
                        ;CLEAR CCR
                        ;DO UNTIL ALL BITS TESTED
                        ;.      WRITE GOOD DATA TO CCR
                        ;.      READ CCR
                        ;.      IF CCR NOT EQUAL TO GOOD DATA THEN
                        ;.      .       IF CCR EQUAL TO ZERO THEN
                        ;.      .       .       IF GOOD DATA NOT EQUAL TO BITS<15-11> OR BIT 8 THEN
                        ;.      .       ERROR IN CACHE SYSTEM
                        ;.      .       .       ENDIF
                        ;.      .       ELSE
                        ;.      .       ERROR IN CACHE SYSTEM
                        ;.      .       ENDIF
                        ;.      ENDIF
                        ;.      UPDATE TO NEXT BIT
                        ;ENDDO
                        ;ENDTST
                        ;;*****************************************************************
002312                  TST2:
002312  005267  176466          INC     $TESTN                  ;INCREMENT TEST NUMBER
002316  012701  000001          MOV     #1,     R1              ;INITIALIZE GOOD DATA TO #1
002322  005067  175420          CLR     CCR                     ;CLEAR CCR
002326  010167  175414  1$:     MOV     R1,     CCR             ;WRITE GOOD DATA TO CCR
002332  026701  175410          CMP     CCR,    R1              ;IF CCR NOT EQUAL GOOD DATA
002336  001415                  BEQ     4$                      ;THEN
002340  005767  175402          TST     CCR                     ;IF CCR EQUAL TO ZERO
002344  001007                  BNE     3$                      ;THEN
002346  032701  174400          BIT     #174400,R1              ;IF GOOD DATA NOT BITS<15-11> OR 8
002352  001007                  BNE     4$                      ;THEN
002354  104000                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
002356  000003                  .WORD   3                       ;UNIQUE ERROR NUMBER
002360  001325                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                ;CACHE SYSTEM ERROR
002362  000403                  BR      4$                      ;ENDIF
002364                  3$:
002364  104000                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
002366  000004                  .WORD   4                       ;UNIQUE ERROR NUMBER
002370  001325                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                ;CACHE SYSTEM ERROR
002372  022737  002000  177746  4$:     CMP     #2000,@#CCR     ;WAS LAST FUNCTION WRITE WRONG TAG PAR?
002400  001003                  BNE     5$                      ;IF NOT GO SHIFT LEFT AND DO NEXT TEST.
002402  012737  000400  177746          MOV     #400,@#CCR      ;FLUSH CACHE TO GET RID OF WRONG
                                                                ;TAG PARITY
002410  006101                  5$:     ROL     R1              ;UPDATE TO NEXT BIT
002412  103345                  BCC     1$                      ;DO UNTIL ALL BITS TESTED
```

```
1077
1078  002414                          CTST3:
1079
1080                                  ;;*****************************************************************************
1081                                  ;*TEST 3          TEST FORCE MISS FUNCTION
1082                                  ;;*****************************************************************************
1083                                  ;FORCE MISS TEST - THIS TEST WILL VERIFY THAT ALL REFERENCES MADE
1084                                  ;WITH EITHER BIT<3> OR BIT<2> OF THE CCR SET CAUSE A CACHE MISS AND
1085                                  ;LEAVE THE CACHE ENTRY UNCHANGED. FIRST WRITE A TEST ADDRESS WITH
1086                                  ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN
1087                                  ;INTO THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH
1088                                  ;NEW DATA. NEXT CLEAR BITS<3:2> AND READ THE TEST ADDRESS, THE DATA
1089                                  ;SHOULD EQUAL TO PATTERN 1. FINALLY READ THE TEST ADDRESS WITH BIT<3>
1090                                  ;SET, THE DATA SHOULD EQUAL PATTERN 2. A LAST WRITE MUST BE DONE WITH
1091                                  ;BOTH FORCE BITS CLEARED BECAUSE THE CACHE AND MAIN MEMORY HAVE
1092                                  ;DIFFERENT DATA.
1093                                  ;
1094                                  ;
1095                                  ;BGNTST
1096                                  ;WRITE TEST ADDRESS WITH PATTERN 1
1097                                  ;SET CCR BITS<3:2> = 0,1
1098                                  ;WRITE TEST ADDRESS WITH PATTERN 2
1099                                  ;CLEAR CCR BIT<3>
1100                                  ;READ TEST ADDRESS
1101                                  ;SAVE HIT/MISS REGISTER DATA
1102                                  ;COMPARE RECEIVED DATA TO PATTERN 1
1103                                  ;IF DATA NOT EQUAL THEN
1104                                  ;.      ERROR IN CACHE SYSTEM
1105                                  ;ENDIF
1106                                  ;SET CCR<3:2> = 1,0
1107                                  ;READ TEST ADDRESS
1108                                  ;SAVE HIT/MISS REGISTER DATA
1109                                  ;COMPARE RECEIVED DATA TO PATTERN 2
1110                                  ;IF DATA NOT EQUAL THEN
1111                                  ;.      ERROR IN CACHE SYSTEM
1112                                  ;ENDIF
1113                                  ;CLEAR CCR<3:2>
1114                                  ;WRITE TEST ADDRESS
1115                                  ;ENDTST
1116                                  ;
1117  002414                          ;;*****************************************************************************
1118  002414  005267  176364          TST3:   INC     $TESTN                  ;INCREMENT TEST NUMBER
1119  002420  012737  002550  000114          MOV     #FMPARR,@#114           ;SETUP PARITY VECTOR TO POINT TO HANDLER
1120  002426  005037  001122                  CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
1121                                                                          ;ALLOCATES CACHE AND MEMORY WITH ZEROS
1122  002432  012767  000004  175306          MOV     #BIT02, CCR             ;SET CCR BITS<3:2> = 0,1(FORCE CACHE MISS)
1123  002440  012737  177777  001122          MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
1124                                                                          ;WRITES MEMORY WITH ONES, CACHE STILL ZEROS
1125  002446  012767  000200  175272          MOV     #200,   CCR             ;CLEAR CCR BIT 3 (CLEAR FORCE MISS)
1126                                                                          ;AND SET PARITY ABORTS
1127  002454  013701  001122                  MOV     @#TSTLOC,R1             ;READ TEST ADDRESS (SHOULD READ ZEROS IN CACHE)
1128  002460  005701                          TST     R1                      ;COMPARE RECEIVED DATA TO PATTERN 1
1129  002462  001403                          BEQ     1$                      ;IF DATAS NOT EQUAL THEN
1130  002464  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1131  002466  000005                          .WORD   5                       ;UNIQUE ERROR NUMBER
1132  002470  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
```

```
1133                                                                    ;CACHE SYSTEM ERROR
1134  002472  052767  000010  175246  1$:     BIS     #BIT03, CCR       ;SET CCR BITS <3:2> = 1,0(FORCE CACHE MISS)
1135  002500  013701  001122            MOV     @#TSTLOC,R1       ;READ TEST LOCATION (SHOULD READ ONES IN MEMORY)
1136  002504  020127  177777            CMP     R1,     #177777   ;COMPARE RECEIVED DATA TO PATTERN 2
1137  002510  001403                    BEQ     2$                ;IF DATAS NOT EQUAL THEN
1138  002512  104000                    ERROR                     ;ALL ERRORS TO TRAP TO EMT VECTOR
1139  002514  000006                    .WORD   6                 ;UNIQUE ERROR NUMBER
1140  002516  001325                    .WORD   ERRMSG            ;ADDRESS OF ERROR MESSAGE
1141                                                                    ;CACHE SYSTEM ERROR
1142  002520  005067  175222    2$:     CLR     CCR               ;CLEAR CCR BITS<3:2>
1143  002524  005037  001122            CLR     @#TSTLOC          ;WRITE TEST ADDRESS
1144  002530  005037  177744            CLR     @#MSER            ;CLEAR THE MEM SYS ERROR REG
1145  002534  005037  177766            CLR     @#CPEREG          ;CLEAR THE CPU ERROR REGISTER
1146  002540  005037  001116            CLR     @#SOFTER          ;CLEAR THE SOFT ERROR COUNT
1147  002544  000167  000006            JMP     TST4              ;JUMP OVER DATA TABLES OR SUBROUTINES
1148                                                              ;TO NEXT TEST
1149
1150  002550                    FMPARR:                           ;PARITY ERROR ON FORCE MISS
1151  002550  104000                    ERROR                     ;ALL ERRORS TO TRAP TO EMT VECTOR
1152  002552  000007                    .WORD   7                 ;UNIQUE ERROR NUMBER
1153  002554  001325                    .WORD   ERRMSG            ;ADDRESS OF ERROR MESSAGE
1154
1155
1156  002556                    CTST4:
1157                            ;;*****************************************************************
1158                            ;*TEST 4          TEST HIT MISS REGISTER
1159                            ;;*****************************************************************
1160                            ;HIT/MISS REGISTER TEST PART 1 - THIS TEST WILL VERIFY THAT THE HIT/MISS
1161                            ;REGISTER CORRECTLY LOGS HITS AND MISSES.FIRST WRITE A TEST ADDRESS WITH
1162                            ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN INTO
1163                            ;THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH NEW DATA.
1164                            ;NEXT CLEAR CCR BITS<3:2> AND READ THE TEST ADDRESS. THE HIT/MISS REGISTER
1165                            ;SHOULD HAVE LOGGED A HIT. FINALLY READ THE TEST ADDRESS PLUS 20000(8) THE
1166                            ;HIT/MISS REGISTER SHOULD HAVE LOGGED A MISS.
1167                            ;
1168                            ;
1169                            ;BGNTST
1170                            ;WRITE TEST ADDRESS WITH PATTERN 1
1171                            ;SET CCR BITS<3:2> = 0,1
1172                            ;WRITE TEST ADDRESS WITH PATTERN 2
1173                            ;CLEAR CCR BIT<3>
1174                            ;READ TEST ADDRESS
1175                            ;IF HIT/MISS REGISTER BIT 3 NOT SET THEN
1176                            ;.      ERROR IN CACHE SYSTEM
1177                            ;ENDIF
1178                            ;READ TEST ADDRESS + 20000(8)
1179                            ;IF HIT/MISS REGISTER BIT 3 SET THEN
1180                            ;.      ERROR IN CACHE SYSTEM
1181                            ;ENDIF
1182                            ;ENDTST
1183                            ;
1184                            ;;*****************************************************************
1185  002556                    TST4:
1186  002556  005267  176222            INC     $TESTN                    ;INCREMENT TEST NUMBER
1187  002562  012737  002702  000114    MOV     #HMPARR,@#114             ;SETUP PARITY VECTOR TO POINT TO HANDLER
1188  002570  005037  001122            CLR     @#TSTLOC                  ;WRITE TEST LOCATION WITH PATTERN 1
```

```
1189
1190  002574  012767  000004  175144          MOV     #BIT02, CCR          ;WRITE CACHE AND MEMORY WITH ZEROS
1191  002602  012737  177777  001122          MOV     #177777,@#TSTLOC     ;SET CCR BITS<3:2> = 0,1(FORCE CACHE MISS)
1192                                                                       ;WRITE TEST LOCATION WITH PATTERN 2
1193  002610  012767  000200  175130          MOV     #200,   CCR          ;MEMORY =-1, CACHE STILL ZERO
1194                                                                       ;CLEAR CCR BIT 3 (CLEAR FORCE MISS)
1195  002616  013701  001122                  MOV     @#TSTLOC,R1          ; AND SET PARITY ABORTS
1196  002622  032737  000010  177752          BIT     #BIT03, @#HITMIS     ;READ TEST ADDRESS (SHOULD BE CACHE HIT)
1197  002630  001003                          BNE     1$                   ;IF HIT/MISS REGISTER BIT 3 NOT SET
1198  002632  104000                          ERROR                        ;THEN
1199  002634  000010                          .WORD   10                   ;ALL ERRORS TO TRAP TO EMT VECTOR
1200  002636  001325                          .WORD   ERRMSG               ;UNIQUE ERROR NUMBER
1201                                                                       ;ADDRESS OF ERROR MESSAGE
1202  002640  013701  021122          1$:     MOV     @#TSTLOC+8192.,R1    ;CACHE SYSTEM ERROR
1203  002644  032737  000010  177752          BIT     #BIT03, @#HITMIS     ;READ TEST LOCATION + 20000(8) SHOULD MISS
1204  002652  001403                          BEQ     2$                   ;IF HIT/MISS REGISTER BIT 3 SET
1205  002654  104000                          ERROR                        ;THEN
1206  002656  000011                          .WORD   11                   ;ALL ERRORS TO TRAP TO EMT VECTOR
1207  002660  001325                          .WORD   ERRMSG               ;UNIQUE ERROR NUMBER
1208                                                                       ;ADDRESS OF ERROR MESSAGE
1209  002662                          2$:                                  ;CACHE SYSTEM ERROR
1210  002662  005037  177744                  CLR     @#MSER               ;CLEAR THE MEM SYS ERROR REG
1211  002666  005037  177766                  CLR     @#CPEREG             ;CLEAR THE CPU ERROR REGISTER
1212  002672  005037  001116                  CLR     @#SOFTER             ;CLEAR THE SOFT ERROR COUNT
1213  002676  000167  000006                  JMP     TST5                 ;JUMP OVER DATA TABLES OR SUBROUTINES
1214                                                                       ;TO NEXT TEST
1215
1216  002702                          HMPARR:                              ;PARITY ERROR DURING HIT/MISS REGISTER TEST
1217  002702  104000                          ERROR                        ;ALL ERRORS TO TRAP TO EMT VECTOR
1218  002704  000012                          .WORD   12                   ;UNIQUE ERROR NUMBER
1219  002706  001325                          .WORD   ERRMSG               ;ADDRESS OF ERROR MESSAGE
1220
1221
1222
1223  002710                          CTST5:
1224                                  ;;*************************************************************************
1225                                  ;*TEST 5        TEST HIT MISS REGISTER BITS
1226                                  ;;*************************************************************************
1227                                  ;HIT/MISS REGISTER TEST - THIS TEST WILL VERIFY THAT THE HIT/MISS
1228                                  ;REGISTER CORRECTLY LOGS CACHE HITS AND MISSES. IT WILL ALSO VERIFY
1229                                  ;THAT EACH BIT OF THE REGISTER IS UNIQUE. THIS WILL BE DONE BY
1230                                  ;FLOATING A ZERO THROUGH A FIELD OF ONES. THE ROTATING WILL BE DONE
1231                                  ;BY EXECUTING A TST @#HM (WHERE HM IS THE ADDRESS OF THE HIT/MISS
1232                                  ;REGISTER) AT SUCCESSIVE POSITIONS IN A SET OF FOUR NOPS. THE READ
1233                                  ;OF THE I/O PAGE ADDRESS OF THE HIT/MISS REGISTER SHOULD CAUSE A
1234                                  ;MISS TO BE RECORDED.
1235                                  ;
1236                                  ;BGNTST
1237                                  ;INITIALIZE LOOP INDICATOR
1238                                  ;INITIALIZE EXPECTED DATA
1239                                  ;DO UNTIL LOOP INDICATOR = SEVEN
1240                                  ;.      PUT BACKGROUND DATA INTO EXECUTION BUFFER
1241                                  ;.      PUT TST INSTRUCTION INTO TEST LOCATION
1242                                  ;.      JUMP TO EXECUTE BUFFER
1243                                  ;.      IF EXPECTED DATA NE RECEIVED DATA THEN
1244                                  ;.      .       ERROR IN CACHE SYSTEM
```

```
1245                                    ;.      ENDIF
1246                                    ;.      INCREMENT LOOP INDICATOR
1247                                    ;.      UPDATE EXPECTED DATA
1248                                    ;ENDDO
1249                                    ;EXIT   TST
1250                                    ;
1251                                    ;BACKGROUND DATA:
1252                                    ;               NOP
1253                                    ;               NOP
1254                                    ;               NOP
1255                                    ;               NOP
1256                                    ;               MOV     @#HM,R5
1257                                    ;               RTS     PC
1258                                    ;ENDTST
1259                                    ;;*****************************************************************************
1260  002710                            TST5:
1261  002710  005267  176070                   INC     $TESTN                  ;INCREMENT TEST NUMBER
1262  002714  012737  001630  000114           MOV     #RAMPAR,@#114           ;SET UP VECTOR FOR UNEXPECTED PARITY ERROR
1263  002722  005067  176132                   CLR     LOOPIN                  ;INITIALIZE LOOP INDICATOR
1264  002726  012703  020000                   MOV     #20000,R3               ;LOAD R3 WITH 1ST ADDR IN 2ND BANK OF MEM.
1265  002732  012704  003070                   MOV     #EXPTBL,R4              ;GET ADDRESS OF EXPECTED DATA TABLE
1266  002736  026727  176116  000007  1$:      CMP     LOOPIN, #7              ;DO UNTIL LOOP INDICATOR EQUALS 7
1267  002744  001432                           BEQ     ENDHRT                  ;THEN EXIT
1268  002746  012705  000007                   MOV     #7,R5                   ;LOAD R2 WITH 7
1269  002752  012701  003052                   MOV     #BACDAT,R1              ;PUT BACKGROUND DATA INTO
1270  002756  012702  001122                   MOV     #TSTLOC,R2              ;EXECUTION BUFFER
1271  002762  012122                   2$:     MOV     (R1)+,  (R2)+
1272  002764  077502                           SOB     R5,     2$              ;LOOP FOR SEVEN WORDS
1273  002766  005763  001122                   TST     TSTLOC(R3)              ;SETUP MISS ON FETCH OF INSTRUCTION
1274  002772  005723                           TST     (R3)+                   ;INC R3 BY 2
1275  002774  004767  176122                   JSR     PC,     TSTLOC          ;GO EXECUTE TEST CODE
1276  003000  022405                           CMP     (R4)+,  R5              ;IF EXPECTED DATA NOT EQUAL TO
1277  003002  001003                           BNE     3$                      ;RECEIVED DATA THEN
1278  003004  005267  176050                   INC     LOOPIN                  ;INCREMENT LOOP COUNTER
1279  003010  000752                           BR      1$                      ;
1280  003012                            3$:
1281  003012  012737  002714  001120           MOV     #TST5+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
1282  003020  004767  176614                   JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
1283  003024  104000                           ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1284  003026  000013                           .WORD   13                      ;UNIQUE ERROR NUMBER
1285  003030  001325                           .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1286                                                                            ;CACHE SYSTEM ERROR
1287  003032                            ENDHRT:
1288  003032  005037  177744                   CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
1289  003036  005037  177766                   CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
1290  003042  005037  001116                   CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
1291  003046  000167  000034                   JMP     TST6                    ;JUMP OVER DATA TABLES OR SUBROUTINES
1292                                                                            ;TO NEXT TEST
1293
1294  003052  000240                    BACDAT: .WORD   NOP
1295  003054  000240                            .WORD   NOP
1296  003056  000240                            .WORD   NOP
1297  003060  000240                            .WORD   NOP
1298  003062  013705  177752                   MOV     @#HITMIS,R5             ;SAVE CONTENTS OF HIT/MISS REGISTER
1299  003066  000207                           RTS     PC
1300
```

```
1301  003070  000077                    EXPTBL: .WORD   77
1302  003072  000037                            .WORD   37
1303  003074  000057                            .WORD   57
1304  003076  000067                            .WORD   67
1305  003100  000073                            .WORD   73
1306  003102  000075                            .WORD   75
1307  003104  000076                            .WORD   76
1308
1309
1310
1311
1312  003106                          CTST6:
1313                                  ;;*************************************************************************
1314                                  ;*TEST 6         TEST BYTE ALLOCATION OF CACHE
1315                                  ;;*************************************************************************
1316                                  ;BYTE ALLOCATION TEST- THIS TEST WILL VERIFY THAT THE CACHE SYSTEM CORRECTLY
1317                                  ;HANDLES BYTE ACCESSES. THE TEST WILL FIRST CHECK THAT A WRITE ACCESS WHICH
1318                                  ;IS A MISS DOES NOT UPDATE THE CACHE. THIS WILL BE DONE BY FIRST ASSURRING
1319                                  ;A MISS AT A TEST LOCATION. THEN A WRITE BYTE ACCESS WILL BE DONE. FINALLY
1320                                  ;THE LOCATION WILL BE READ. THE WRITE BYTE SHOULD NOT HAVE ALLOCATED THE
1321                                  ;ADDRESS. NEXT THE TEST WILL VERIFY THAT IF A WRITE BYTE ACCESS IS A HIT
1322                                  ;THAT THE CACHE LOCATION IS UPDATED. THE TEST WILL FIRST WRITE A TEST LOCATION
1323                                  ;WITH A PATTERN TO ALLOCATE THE ADDRESS. THEN A SECOND PATTERN WILL BE WRITTEN
1324                                  ;TO THE HIGH BYTE OF THE TEST LOCATION. THE TEST LOCATION WILL THEN BE READ.
1325                                  ;IF THE HIGH BYTE OF THE TEST LOCATION IS EQUAL TO THE SECOND PATTERN THEN THE
1326                                  ;LOCATION WAS UPDATED. THE SECOND TEST WILL BE REPEATED TO TEST LOW BYTE
1327                                  ;ACCESSES.
1328                                  ;
1329                                  ;BGNTST
1330                                  ;SAVE VECTOR 114
1331                                  ;LET VECTOR 114 POINT TO BYTE PARITY ROUTINE
1332                                  ;SET PARITY ABORT BIT IN CACHE CONTROL REGISTER
1333                                  ;READ TEST LOCATION + 4K
1334                                  ;WRITE LOW BYTE TO TEST ADDRESS
1335                                  ;READ LOW BYTE OF TEST ADDRESS
1336                                  ;IF HIT/MISS REGISTER BIT 3 SET THEN
1337                                  ;.      ERROR IN CACHE SYSTEM
1338                                  ;ENDIF
1339                                  ;WRITE PATTERN 1 TO TEST LOCATION
1340                                  ;WRITE BYTE PATTERN 2 TO TEST LOCATION+1
1341                                  ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
1342                                  ;.      ERROR IN CACHE SYSTEM
1343                                  ;ELSE
1344                                  ;.      READ TEST LOCATION
1345                                  ;.      IF RECEIVED DATA NOT EQUAL TO EXPECTED DATA THEN
1346                                  ;.      .       ERROR IN CACHE SYSTEM
1347                                  ;ENDIF
1348                                  ;WRITE PATTERN 1 TO TEST LOCATION
1349                                  ;WRITE BYTE PATTERN 2 TO TEST LOCATION LOW BYTE
1350                                  ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
1351                                  ;.      ERROR IN CACHE SYSTEM
1352                                  ;ELSE
1353                                  ;.      READ TEST LOCATION
1354                                  ;.      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
1355                                  ;.      .       ERROR IN CACHE SYSTEM
1356                                  ;ENDIF
```

```
                                        ;CLEAR CACHE CONTROL REGISTER
                                        ;RESTORE VECTOR 114
                                        ;EXIT   TST
                                        ;
                                        ;BYTE PARITY ROUTINE:   SAVE MSER
                                        ;                       RETURN
                                        ;ENDTST
                                        ;;*****************************************************************************
003106                                  TST6:
003106  005267  175672                          INC     $TESTN                  ;INCREMENT TEST NUMBER
003112  012737  003374  000114                  MOV     #BYPARR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
003120  012767  000200  174620                  MOV     #BIT07, CCR             ;SET PARITY ABORT BIT IN CCR
003126  005767  015770                          TST     TSTLOC+8192.            ;READ TEST LOCATION + 4K TO CAUSE MISS
                                                                                ;ON NEXT ACCESS OF TSTLOC
003132  105067  175764                          CLRB    TSTLOC                  ;WRITE LOW BYTE OF TEST LOCATION
003136  105767  175760                          TSTB    TSTLOC                  ;READ LOW BYTE OF TEST LOCATION
003142  032737  000010  177752                  BIT     #BIT03, @#HITMIS        ;IF BIT3 OF HIT/MISS REGISTER NOT SET
003150  001410                                  BEQ     1$                      ;THEN BRANCH
003152  012737  003112  001120                  MOV     #TST6+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
003160  004767  176454                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
003164  104000                                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
003166  000014                                  .WORD   14                      ;UNIQUE ERROR NUMBER
003170  001325                                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                        ;CACHE SYSTEM ERROR
003172  005067  175724                  1$:     CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
003176  152737  000377  001123                  BISB    #377,   @#TSTLOC+1      ;WRITE PATTERN 2 TO HIGH BYTE
003204  032767  000010  174540                  BIT     #BIT03, HITMIS          ;IF BIT3 NOTSETIN HIT/MISS REGISTER
003212  001011                                  BNE     2$                      ;THEN
003214  012737  003112  001120                  MOV     #TST6+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
003222  004767  176412                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
003226  104000                                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
003230  000015                                  .WORD   15                      ;UNIQUE ERROR NUMBER
003232  001325                                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                        ;CACHE SYSTEM ERROR
003234  000414                                  BR      3$                      ;ELSE
003236  022737  177400  001122          2$:     CMP     #177400,@#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
003244  001410                                  BEQ     3$                      ;PATTERN 2 THEN
003246  012737  003112  001120                  MOV     #TST6+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
003254  004767  176360                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
003260  104000                                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
003262  000016                                  .WORD   16                      ;UNIQUE ERROR NUMBER
003264  001325                                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                        ;CACHE SYSTEM ERROR
003266  005067  175630                  3$:     CLR     TSTLOC                  ;WRITE PATTERN 1 TO TEST LOCATION
003272  152737  000377  001122                  BISB    #377,   @#TSTLOC        ;WRITE PATTERN 2 TO LOW BYTE
003300  032767  000010  174444                  BIT     #BIT03, HITMIS          ;IF BIT03 SET IN HIT/MISS REGISTER
003306  001011                                  BNE     4$                      ;THEN BRANCH; WE GOT A HIT.
                                                                                ;ELSE
003310  012737  003112  001120                  MOV     #TST6+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
003316  004767  176316                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
003322  104000                                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
003324  000017                                  .WORD   17                      ;UNIQUE ERROR NUMBER
003326  001325                                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                        ;CACHE SYSTEM ERROR
003330  000407                                  BR      5$
003332  022737  000377  001122          4$:     CMP     #377,   @#TSTLOC        ;IF TEST LOCATION NOT EQUAL TO
```

```
1413  003340  001403                        BEQ     5$                      ;PATTERN 2 THEN
1414  003342  104000                        ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1415  003344  000020                        .WORD   20                      ;UNIQUE ERROR NUMBER
1416  003346  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1417                                                                                ;CACHE SYSTEM ERROR
1418  003350  005067  174372        5$:     CLR     CCR                     ;CLEAR CCR BEFORE EXIT
1419  003354  005037  177744                CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
1420  003360  005037  177766                CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
1421  003364  005037  001116                CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
1422  003370  000167  000006                JMP     TST7                    ;JUMP OVER DATA TABLES OR SUBROUTINES
1423                                                                        ;TO NEXT TEST
1424
1425  003374                        BYPARR:                                 ;PARITY ERROR DURING BYTE ALLOCATION TEST
1426  003374  104000                        ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1427  003376  000021                        .WORD   21                      ;UNIQUE ERROR NUMBER
1428  003400  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1429
1430
1431
1432  003402                        CTST7:
1433                                ;;*******************************************************************
1434                                ;*TEST 7        TEST CACHE BYPASS (MMU)
1435                                ;;*******************************************************************
1436                                ;PDR BIT15 (BYPASS) TEST - THIS TEST WILL VERIFY THAT WHEN BIT<15> IS SET
1437                                ;IN A PDR AND AN ACCESS IS MADE TO A LOCATION MAPPED BY THE SELECTED PDR
1438                                ;THAT WOULD NORMALLY CAUSE A CACHE ACCESS, THE CACHE IS BYPASSED (I.E. THE
1439                                ;CACHE LOCATION IS INVALIDATED AND A MAIN MEMORY IS READ. THIS WILL BE DONE
1440                                ;BY FIRST WRITING A TEST LOCATION WITH A KNOWN PATTERN TO ALLOCATE THE ADDRESS.
1441                                ;THEN THE LOCATION WILL BE WRITTEN AGAIN WITH THE MMU ENABLED AND BIT <15> SET
1442                                ;IN THE CONTROLING PDR WITH A NEW PATTERN. THE LOCATION WILL THEN BE READ
1443                                ;AND A MISS SHOULD BE LOGGED IN THE HIT/MISS REGISTER. THIS READ WILL ALSO
1444                                ;BRING VALID DATA INTO CACHE FROM MEMORY. THE MMU WILL AGAIN BE ENABLED
1445                                ;WITH BIT <15> SET AND A READ CYCLE DONE. THIS SHOULD INVALIDATE THE CACHE.
1446                                ;NEXT THE MMU WILL BE DISABLED AND THE LOCATION READ FOR A THIRD TIME. THIS
1447                                ;WILL BE DONE WITH BIT<15> STILL SET. A MISS SHOULD ALSO BE LOGGED. LASTLY
1448                                ;BIT<15> WILL BE CLEARED AND THE MMU ENABLED AND THE LOCATION AGAIN READ.
1449                                ;THIS TIME A CACHE HIT SHOULD BE LOGGED.
1450                                ;
1451                                ;
1452                                ;BGNTST
1453                                ;SETUP MMU REGISTERS
1454                                ;WRITE TEST LOCATION WITH PATTERN 1
1455                                ;SET BIT<15> IN PDR FOR TEST LOCATION
1456                                ;ENABLE MMU
1457                                ;WRITE TEST LOCATION WITH PATTERN 2
1458                                ;DISABLE MMU AND CLEAR BIT<15> IN PDR
1459                                ;READ TEST LOCATION
1460                                ;IF HIT/MISS REGISTER BIT<1> SET THEN
1461                                ;.      ERROR IN CACHE SYSTEM
1462                                ;ENDIF
1463                                ;SET BIT<15> IN PDR FOR TEST LOCATION
1464                                ;ENABLE MMU
1465                                ;READ TEST LOCATION (SHOULD INVALIDATE CACHE)
1466                                ;DISABLE MMU
1467                                ;READ TEST LOCATION
1468                                ;IF HIT/MISS REGISTER BIT<1> SET THEN
```

```
1469                                    ;.      ERROR IN CACHE SYSTEM
1470                                    ;ENDIF
1471                                    ;CLEAR BIT<15> IN PDR
1472                                    ;TURN ON MMU
1473                                    ;READ TEST LOCATION
1474                                    ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
1475                                    ;.      ERROR IN CACHE SYSTEM
1476                                    ;ENDIF
1477                                    ;TURN OFF MMU
1478                                    ;ENDTST
1479                                    ;
1480                                    ;;*************************************************************************
1481  003402                            TST7:
1482  003402  005267  175376                   INC     $TESTN                  ;INCREMENT TEST NUMBER
1483  003406  012737  001630  000114           MOV     #RAMPAR,@#114           ;SET UP FOR UNEXPECTED PARITY ERROR
1484  003414  004767  175772                   JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMEMT REGISTERS
1485  003420  005037  001122                   CLR     @#TSTLOC                ;WRITE TEST LOCATION WITH PATTERN 1
1486  003424  052767  100000  166646           BIS     #BIT15, KIPDR0          ;SET BIT15 IN PDR FOR TEST LOCATION
1487  003432  052737  000001  177572           BIS     #BIT00,@#SR0            ;TURN ON MEMORY MANAGEMENT
1488  003440  012737  177777  001122           MOV     #177777,@#TSTLOC        ;WRITE TEST LOCATION WITH PATTERN 2
1489  003446  005067  174120                   CLR     SR0                     ;TURN OFF MEMORY MANAGEMENT
1490  003452  042767  100000  166620           BIC     #BIT15, KIPDR0          ;CLEAR BIT15 IN PDR FOR TEST LOCATION
1491  003460  013701  001122                   MOV     @#TSTLOC,R1             ;READ TEST LOCATION
1492  003464  032767  000010  174260           BIT     #BIT03, HITMIS          ;IF HIT/MISS REGISTER BIT 3 SET
1493  003472  001410                           BEQ     2$                      ;THEN
1494  003474  012737  003406  001120           MOV     #TST7+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
1495  003502  004767  176132                   JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
1496  003506  104000                           ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1497  003510  000022                           .WORD   22                      ;UNIQUE ERROR NUMBER
1498  003512  001325                           .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1499                                                                            ;CACHE SYSTEM ERROR
1500  003514  052767  100000  166556  2$:      BIS     #BIT15, KIPDR0          ;SET BIT15 IN PDR FOR TEST LOCATION
1501  003522  052737  000001  177572           BIS     #BIT00,@#SR0            ;TURN ON MEMORY MANAGEMENT
1502  003530  005767  175366                   TST     TSTLOC                  ;READ TEST LOCATION, BYPASS CACHE
1503  003534  042767  100000  166536           BIC     #BIT15, KIPDR0          ;TURN OFF CACHE BYPASS ON KERNEL PAGE 0.
1504  003542  005067  174024                   CLR     SR0                     ;TURN OFF MMU
1505  003546  005737  001122                   TST     @#TSTLOC                ;READ TEST LOCATION-SHOULD BE CACHE MISS
1506  003552  032767  000010  174172           BIT     #BIT03, HITMIS          ;IF HIT/MISS REGISTER BIT 3 SET
1507  003560  001410                           BEQ     3$                      ;THEN
1508  003562  012737  003406  001120           MOV     #TST7+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
1509  003570  004767  176044                   JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
1510  003574  104000                           ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1511  003576  000023                           .WORD   23                      ;UNIQUE ERROR NUMBER
1512  003600  001325                           .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1513                                                                            ;CACHE SYSTEM ERROR
1514  003602  052737  000001  177572  3$:      BIS     #BIT00,@#SR0            ;TURN ON . .MORY MANAGEMENT
1515  003610  005737  001122                   TST     @#TSTLOC                ;READ TEST LOCATION ONE MORE TIME.
1516                                                                            ;PREVIOUS READ SHOULD HAVE ALLOCATED
1517                                                                            ;CACHE SO WE'RE EXPECTING A CACHE HIT
1518  003614  032767  000010  174130           BIT     #BIT03, HITMIS          ;IF HIT/MISS REGISTER BIT 3 NOT SET
1519  003622  001012                           BNE     4$                      ;THEN
1520  003624  005067  173742                   CLR     SR0                     ;TURN OFF MMU BEFORE ERROR ROUTINE.
1521  003630  012737  003406  001120           MOV     #TST7+4,@#SOFTRE        ;MOVE RETRY ADDRESS TO SOFT RETURN
1522  003636  004767  175776                   JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
1523  003642  104000                           ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1524  003644  000024                           .WORD   24                      ;UNIQUE ERROR NUMBER
```

```
1525  003646  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1526                                                                          ;CACHE SYSTEM ERROR
1527  003650  042767  000001  173714  4$:     BIC     #BIT00, @#SR0           ;TURN OFF MMU
1528  003656  005037  177744                  CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
1529  003662  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
1530  003666  005037  001116                  CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
1531  003672  000167  000000                  JMP     TST10                   ;JUMP OVER DATA TABLES OR SUBROUTINES
1532                                                                          ;TO NEXT TEST
1533
1534
1535  003676                          CTST10:
1536                                  ;;*************************************************************************
1537                                  ;*TEST 10        TEST CACHE FLUSH
1538                                  ;;*************************************************************************
1539                                  ;FLUSH CACHE TEST - THIS TEST WILL VERIFY THAT WHEN CCR BIT<8> IS
1540                                  ;SET, THE ENTIRE CACHE IS INVALIDATED. FIRST 8K BYTES OF ADDRESSES
1541                                  ;WILL BE READ TO ALLOCATE CACHE. THEN THE CACHE WILL BE FLUSHED.
1542                                  ;THE SAME SET OF ADDRESS WILL THEN BE READ AND THE HIT/MISS REGISTER
1543                                  ;CHECKED AFTER EACH READ FOR A MISS TO BE LOGGED.
1544                                  ;
1545                                  ;BGNTST
1546                                  ;GET FIRST ADDRESS OF 8KB BUFFER
1547                                  ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
1548                                  ;DO UNTIL LOOP COUNTER = 0
1549                                  ;.      READ @ADDRESS+
1550                                  ;ENDDO
1551                                  ;GET FIRST ADDRESS OF 8KB BUFFER
1552                                  ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
1553                                  ;INITIALIZE MISS COUNT
1554                                  ;FLUSH CACHE
1555                                  ;DO UNTIL LOOP COUNTER = 0
1556                                  ;.      READ @ADDRESS+
1557                                  ;.      IF HIT/MISS REGISTER BIT<1> SET THEN
1558                                  ;.      .       DECREMENT MISS COUNT
1559                                  ;.      ENDIF
1560                                  ;ENDDO
1561                                  ;IF MISS COUNT NOT EQUAL TO 4096. THEN
1562                                  ;.      ERROR IN CACHE SYSTEM
1563                                  ;ENDIF
1564                                  ;ENDTST
1565                                  ;
1566                                  ;;*************************************************************************
1567  003676                          TST10:
1568  003676  005267  175102                  INC     $TESTN                  ;INCREMENT TEST NUMBER
1569  003702  004767  175504                  JSR     PC,     INITMM          ;INITIALIZE MMU
1570  003706  005067  166442                  CLR     KIPAR6                  ;LET PAR6 POINT TO LOW 4K
1571  003712  012701  140000                  MOV     #140000,R1              ;GET ADDRESS OF 8K BYTE BUFFER
1572  003716  012702  010000                  MOV     #4096., R2              ;INIT LOOP COUNTER TO 4K WORD COUNT
1573  003722  052737  000001  177572          BIS     #BIT00,@#SR0            ;ENABLE MMU
1574  003730  005721                  1$:     TST     (R1)+                   ;READ ADDRESS TO ALLOCATE CACHE
1575  003732  077202                          SOB     R2,     1$              ;DO UNTIL ALL LOCATIONS ALLOCATED
1576  003734  012701  140000                  MOV     #140000,R1              ;GET ADDRESS OF 8K BYTE BUFFER
1577  003740  012702  010000                  MOV     #4096., R2              ;INIT LOOP COUNTER TO 4K WORD COUNT
1578  003744  012703  010000                  MOV     #4096., R3              ;PRE-LOAD MISS COUNT WITH 4096.
1579  003750  012767  000400  173770          MOV     #400,   CCR             ;FLUSH CACHE
1580  003756  005721                  2$:     TST     (R1)+                   ;READ ADDRESS
```

```
1581  003760  032737  000010  177752          BIT     #BIT03,@#HITMIS         ;SHOULDN'T GET HITS AT (R1)
1582                                                                          ;IF BIT03 IN HIT/MISS REG INDICATES A MISS
1583                                                                          ;THEN
1584  003766  001401                          BEQ     3$                      ;BRANCH TO 3$ --DON'T DECREMENT THE MISS COUNT
1585                                                                          ;ELSE
1586  003770  005303                          DEC     R3                      ;DECREMENT MISS COUNT IF WE GET A HIT
1587  003772  077207                  3$:     SOB     R2,     2$              ;LOOP UNTIL ENTIRE CACHE CHECKED
1588                                                                          ;FALLING THROUGH SOB INDICATES THAT ALL
1589                                                                          ;CACHE LOCATIONS HAVE BEEN CHECKED.
1590  003774  005067  173572                  CLR     SR0                     ;TURN OFF MMU
1591  004000  022703  010000                  CMP     #4096., R3              ;IF MISS COUNT NOT EQUAL TO 4096.
1592  004004  001410                          BEQ     5$                      ;THEN IT INDICATES THAT NOT ALL CACHE
1593                                                                          ;LOCATIONS WERE FLUSHED
1594
1595  004006  012737  003702  001120          MOV     #TST10+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
1596  004014  004767  175620                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
1597  004020  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1598  004022  000025                          .WORD   25                      ;UNIQUE ERROR NUMBER
1599  004024  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1600                                                                                  ;CACHE SYSTEM ERROR
1601  004026  005037  177572          5$:     CLR     @#SR0                   ;TURN OFF MMU
1602  004032  005037  177744                  CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
1603  004036  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
1604  004042  005037  001116                  CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
1605  004046  000167  000000                  JMP     TST11                   ;JUMP OVER DATA TABLES OR SUBROUTINES
1606                                                                          ;TO NEXT TEST
1607
1608
1609  004052                          CTST11:
1610                                  ;;*****************************************************************
1611                                  ;*TEST 11       TEST UNCONDITIONAL CACHE BYPASS
1612                                  ;;*****************************************************************
1613                                  ;UNCONDITIONAL BYPASS TEST - THIS TEST WILL VERIFY THAT WHEN CCR
1614                                  ;BIT<9> IS SET, A MEMORY REFERENCE IS FORCED TO MAIN MEMORY. THIS
1615                                  ;WILL ALSO VERIFY THAT READ AND WRITE HIT INVALIDATE THE CACHE
1616                                  ;LOCATIONS WHEN BYPASS IS SET.
1617                                  ;
1618                                  ;BGNTST
1619                                  ;READ TEST LOCATIONS TO SETUP POSSIBILITY OF HIT
1620                                  ;SET CCR BIT<9>
1621                                  ;READ FIRST TEST LOCATION
1622                                  ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
1623                                  ;.      ERROR IN CACHE SYSTEM
1624                                  ;ENDIF
1625                                  ;WRITE SECOND TEST LOCATION
1626                                  ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
1627                                  ;.      ERROR IN CACHE SYSTEM
1628                                  ;ENDIF
1629                                  ;CLEAR CCR BIT<9>
1630                                  ;READ FIRST LOCATION
1631                                  ;IF HIT/MISS REGISTER BIT<1> SET THEN
1632                                  ;.      ERROR IN CACHE SYSTEM
1633                                  ;ENDIF
1634                                  ;READ SECOND LOCATION
1635                                  ;IF HIT/MISS REGISTER BIT<1> SET THEN
1636                                  ;.      ERROR IN CACHE SYSTEM
```

```
                                        ;ENDIF
                                        ;ENDTST
                                        ;
                                        ;;*****************************************************************
004052                                  TST11:
004052  005267  174726                          INC     $TESTN                  ;INCREMENT TEST NUMBER
004056  005737  001122                          TST     @#TSTLOC                ;ALLOCATE FIRST TEST LOCATION
004062  005737  001124                          TST     @#TSTLOC+2              ;ALLOCATE SECOND TEST LOCATION
004066  052767  001000  173652                  BIS     #BIT09, CCR             ;SET CCR BIT 9 (CACHE BYPASS)
004074  005737  001122                  1$:     TST     @#TSTLOC                ;READ FIRST TEST LOCATION
004100  032767  000010  173644                  BIT     #BIT03, HITMIS          ;IF HIT/MISS REG. BIT 3 NOT SET
004106  001010                                  BNE     2$                      ;THEN
004110  012737  004056  001120                  MOV     #TST11+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
004116  004767  175516                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
004122  104000                                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
004124  000026                                  .WORD   26                      ;UNIQUE ERROR NUMBER
004126  001325                                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                        ;CACHE SYSTEM ERROR
004130  005037  001124                  2$:     CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
004134  013703  177752                  100$:   MOV     @#HITMIS,R3             ;SAVE CONTENTS OF HIT MISS
004140  012704  004134                          MOV     #100$,R4                ;MOV ADDR OF MOV INST TO R4
004144  012702  001124                          MOV     #TSTLOC+2,R2            ;
004150  042704  177701                          BIC     #177701,R4              ;STRIP OFF UNNEEDED BITS
004154  042702  177701                          BIC     #177701,R2              ;
004160  020204                                  CMP     R2,R4                   ;ARE THEY EQUAL?
004162  001004                                  BNE     101$                    ;BRANCH IF NOT
004164  032703  000010                          BIT     #BIT03, R3              ;IF HIT/MISS REG. BIT 3 NOT SET
004170  001014                                  BNE     3$                      ;THEN
004172  000403                                  BR      102$                    ;IT WAS A MISS, GO TO ERROR
004174  032703  000004                  101$:   BIT     #BIT02,R3               ;TEST BIT02
004200  001010                                  BNE     3$                      ;BRANCH IIF IT WAS A HIT
004202                                  102$:
004202  012737  004056  001120                  MOV     #TST11+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
004210  004767  175424                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
004214  104000                                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
004216  000027                                  .WORD   27                      ;UNIQUE ERROR NUMBER
004220  001325                                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                        ;CACHE SYSTEM ERROR
004222  005737  001122                  3$:     TST     @#TSTLOC                ;READ FIRST TEST LOCATION
004226  032767  000010  173516                  BIT     #BIT03, HITMIS          ;IF HIT/MISS REG. BIT 3 SET
004234  001410                                  BEQ     4$                      ;THEN
004236  012737  004056  001120                  MOV     #TST11+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
004244  004767  175370                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
004250  104000                                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
004252  000030                                  .WORD   30                      ;UNIQUE ERROR NUMBER
004254  001325                                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                        ;CACHE SYSTEM ERROR
004256  005037  001124                  4$:     CLR     @#TSTLOC+2              ;WRITE SECOND LOCATION
004262  016703  173464                  200$:   MOV     HITMIS,R3               ;SAVE CONTENTS OF HIT MISS REG
004266  012704  004262                          MOV     #200$,R4                ;
004272  012702  001124                          MOV     #TSTLOC+2,R2            ;
004276  042704  177701                          BIC     #177701,R4              ;
004302  042702  177701                          BIC     #177701,R2              ;
004306  020204                                  CMP     R2,R4                   ;ARE THEY EQUAL
004310  001004                                  BNE     201$                    ;BRANCH IF NOT
004312  032703  000010                          BIT     #BIT03, R3              ;IF HIT/MISS REG. BIT 3 NOT SET
```

```
1693  004316  001414                          BEQ     5$                      ;THEN EXIT TEST
1694  004320  000403                          BR      202$                    ;ELSE;EXPECTED A MISS GO TO ERROR
1695  004322  032703  000004          201$:   BIT     #BIT02,R3               ;IF HIT/MISS REG. BIT 2 NOT SET
1696  004326  001410                          BEQ     5$                      ;THEN EXIT TEST.ELSE GO TO ERROR
1697  004330                          202$:
1698  004330  012737  004056  001120          MOV     #TST11+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
1699  004336  004767  175276                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
1700  004342  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1701  004344  000031                          .WORD   31                      ;UNIQUE ERROR NUMBER
1702  004346  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1703                                                                          ;CACHE SYSTEM ERROR
1704  004350  005067  173372          5$:     CLR     CCR                     ;CLEAR BYPASS BIT BEFORE EXIT
1705  004354  005037  177744                  CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
1706  004360  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
1707  004364  005037  001116                  CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
1708  004370  000167  000000                  JMP     TST12                   ;JUMP OVER DATA TABLES OR SUBROUTINES
1709                                                                          ;TO NEXT TEST
1710
1711  004374                          CTST12:
1712                                  ;;*********************************************************************
1713                                  ;*TEST 12       TEST WRITE WRONG DATA PARITY
1714                                  ;;*********************************************************************
1715                                  ;WRITE WRONG DATA PARITY TEST- THIS TEST WILL VERIFY THAT WHEN CCR
1716                                  ;BIT<6> = 1 AND CCR BITS<7,0> = 0,1, A READ MISS OCCURS AFTER A
1717                                  ;WRITE. THE WRITE WITH CCR BIT<6> = 1 TO A LOCATION WILL CAUSE A
1718                                  ;CACHE UPDATE AND WRONG PARITY TO BE WRITTEN SO WHEN THE LOCATION
1719                                  ;IS READ INSTEAD OF A HIT BEING RECORDED THE CACHE PARITY ERROR
1720                                  ;WILL CAUSE A CACHE MISS. THIS TEST WILL BE DONE A BYTE AT A TIME.
1721                                  ;NOTE: IF BOTH ERRORS OCCUR CCGA MOST LIKELY BAD.
1722                                  ;       IF ONLY ONE ERROR OCCURS CCGA LEAST LIKELY.
1723                                  ;
1724                                  ;BGNTST
1725                                  ;SAVE CONTENTS OF VECTOR 114
1726                                  ;LET VECTOR 114 POINT TO ABORT ROUTINE
1727                                  ;CLEAR MSER
1728                                  ;IF MSER NOT CLEAR THEN
1729                                  ;.      ERROR IN CACHE SYSTEM
1730                                  ;ENDIF
1731                                  ;WRITE TEST LOCATION
1732                                  ;SET BITS<6,0> IN CCR
1733                                  ;WRITE TEST LOCATION LOW BYTE
1734                                  ;CLEAR CCR BIT<6> (WRITE WRONG DATA PARITY)
1735                                  ;INITIALIZE ERROR INDICATORS
1736                                  ;READ TEST LOCATION LOW BYTE
1737                                  ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
1738                                  ;.      ERROR IN CACHE SYSTEM
1739                                  ;ELSE
1740                                  ;.      IF MSER BITS <7:5> ZERO THEN
1741                                  ;.      ERROR IN CACHE SYSTEM
1742                                  ;.      ENDIF
1743                                  ;ENDIF
1744                                  ;CLEAR MSER
1745                                  ;IF MSER NOT CLEAR THEN
1746                                  ;.      .       ERROR IN CACHE SYSTEM
1747                                  ;ENDIF
1748                                  ;SET CCR BIT<6>
```

```
                                ;WRITE TEST LOCATION HIGH BYTE
                                ;CLEAR CCR BIT<6>
                                ;READ TEST LOCATION HIGH BYTE
                                ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ELSE
                                ;.      IF MSER BITS <7:5> ZERO THEN
                                ;.      .       ERROR IN CACHE SYSTEM
                                ;.      ENDIF
                                ;ENDIF
                                ;RESTORE CONTENTS OF VECTOR 114
                                ;IF ERROR INDICATORS SET THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ENDIF
                                ;EXIT   TST
                                ;
                                ;DATA PARITY ABORT ROUTINE:
                                ;.      ERROR IN CACHE SYSTEM
                                ;                               RETURN
                                ;
                                ;ENDTST
                                ;;*********************************************************************
004374                          TST12:
004374  005267  174404                  INC     $TESTN                  ;INCREMENT TEST NUMBER
004400  013767  000114  174422          MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
004406  012737  004706  000114          MOV     #DAPABO,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
004414  005067  173324                  CLR     MSER                    ;CLEAR MSER
004420  005767  173320                  TST     MSER                    ;IF MSER NOT CLEAR
004424  001403                          BEQ     1$                      ;THEN
004426  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
004430  000032                          .WORD   32                      ;UNIQUE ERROR NUMBER
004432  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                        ;CACHE SYSTEM ERROR
004434  005037  001122          1$:     CLR     @#TSTLOC                ;WRITE TEST LOCATION TO ALLOCATE CACHE
004440  012767  000101  173300          MOV     #101,   CCR             ;SET BITS<6,0> IN CCR
                                                                        ;WRITE WRONG DATA PARITY; IGNORE PARITY INTERRUP
004446  112767  000377  174446          MOVB    #377,   TSTLOC          ;WRITE LOW BYTE WITH BAD PARITY
004454  042767  000100  173264          BIC     #BIT06, CCR             ;CLEAR WRITE WRONG DATA PARITY BIT
004462  105737  001122                  TSTB    @#TSTLOC                ;READ LOW BYTE OF TEST LOCATION
004466  032767  000010  173256          BIT     #BIT03, HITMIS          ;EXPECT A MISS; IF BIT 3 SET IN HIT/MISS REGISTE
004474  001410                          BEQ     2$                      ;THEN
004476  012737  004400  001120          MOV     #TST12+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
004504  004767  175130                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
004510  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
004512  000033                          .WORD   33                      ;UNIQUE ERROR NUMBER
004514  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                        ;CACHE SYSTEM ERROR
004516  032767  000340  173220  2$:     BIT     #340,   MSER            ;IF MSER BITS <7:5> ZERO
004524  001003                          BNE     3$                      ;THEN
004526  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
004530  000034                          .WORD   34                      ;UNIQUE ERROR NUMBER
004532  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                        ;CACHE SYSTEM ERROR
004534  005067  173204          3$:     CLR     MSER                    ;CLEAR MSER
004540  005767  173200                  TST     MSER                    ;IF MSER NOT CLEAR
004544  001403                          BEQ     4$                      ;THEN
```

```
1805  004546  104000                        ERROR
1806  004550  000035                        .WORD   35              ;ALL ERRORS TO TRAP TO EMT VECTOR
1807  004552  001325                        .WORD   ERRMSG          ;UNIQUE ERROR NUMBER
1808                                                                ;ADDRESS OF ERROR MESSAGE
1809  004554  052767  000100  173164  4$:   BIS     #BIT06, CCR     ;CACHE SYSTEM ERROR
1810  004562  112737  000377  001123        MOVB    #377,   @#TSTLOC+1      ;SET CCR BIT 6 (WRITE WRONG PARITY)
1811  004570  042767  000100  173150        BIC     #BIT06, CCR     ;WRITE HIGH BYTE OF TEST LOCATION
1812  004576  105737  001123                TSTB    @#TSTLOC+1      ;CLEAR WRITE WRONG PARITY BIT
1813  004602  032767  000010  173142        BIT     #BIT03, HITMIS  ;READ HIGH BYTE OF TEST LOCATION
1814  004610  001411                        BEQ     5$              ;IF BIT 3 SET IN HIT/MISS REGISTER
1815  004612  012737  004400  001120        MOV     #TST12+4,@#SOFTRE       ;THEN
1816  004620  004767  175014                JSR     PC,APTSFT       ;MOVE RETRY ADDRESS TO SOFT RETURN
1817  004624  104000                        ERROR                   ;GO TO SOFT ERROR SUBROUTINE
1818  004626  000036                        .WORD   36              ;ALL ERRORS TO TRAP TO EMT VECTOR
1819  004630  001325                        .WORD   ERRMSG          ;UNIQUE ERROR NUMBER
1820                                                                ;ADDRESS OF ERROR MESSAGE
1821  004632  000407                        BR      6$              ;CACHE SYSTEM ERROR
1822  004634  032767  000340  173102  5$:   BIT     #340,   MSER    ;GO TO NEXT TEST IF NO HALT ON ERROR
1823  004642  001003                        BNE     6$              ;IF BITS <7:5> ZERO
1824  004644  104000                        ERROR                   ;THEN
1825  004646  000037                        .WORD   37              ;ALL ERRORS TO TRAP TO EMT VECTOR
1826  004650  001325                        .WORD   ERRMSG          ;UNIQUE ERROR NUMBER
1827                                                                ;ADDRESS OF ERROR MESSAGE
1828  004652  012767  000400  173066  6$:   MOV     #BIT08,CCR      ;CACHE SYSTEM ERROR
1829  004660  016737  174144  000114        MOV     SLOC00, @#114   ;FLUSH CACHE, CLEAR CCR BEFORE EXIT
1830  004666                          7$:                           ;RESTORE CONTENTS OF VECTOR 114
1831  004666  005037  177744                CLR     @#MSER          ;CLEAR THE MEM SYS ERROR REG
1832  004672  005037  177766                CLR     @#CPEREG        ;CLEAR THE CPU ERROR REGISTER
1833  004676  005037  001116                CLR     @#SOFTER        ;CLEAR THE SOFT ERROR COUNT
1834  004702  000167  000010                JMP     TST13           ;JUMP OVER DATA TABLES OR SUBROUTINES
1835                                                                ;TO NEXT TEST
1836
1837  004706                          DAPABO:
1838  004706  104000                        ERROR                   ;ALL ERRORS TO TRAP TO EMT VECTOR
1839  004710  000040                        .WORD   40              ;UNIQUE ERROR NUMBER
1840  004712  001325                        .WORD   ERRMSG          ;ADDRESS OF ERROR MESSAGE
1841                                                                ;CACHE SYSTEM ERROR
1842  004714  000002                        RTI
1843
1844
1845
1846  004716                          CTST13:
1847                                  ;;*************************************************************
1848                                  ;*TEST 13       TEST WRITE WRONG TAG PARITY
1849                                  ;;*************************************************************
1850                                  ;WRITE WRONG TAG PARITY - THIS TEST WILL VERIFY THAT A READ MISS
1851                                  ;OCCURS AFTER A WRITE WILL CCR<10> = 1 AND CCR<7,0> = 0,1. THE WRITE
1852                                  ;TO THE LOCATION WILL CAUSE A CACHE UPDATE BUT THE TAG WILL BE
1853                                  ;WRITTEN WITH THE WRONG PARITY. WHEN THE LOCATION IS READ INSTEAD
1854                                  ;OF A CACHE HIT OCCURRING THE PARITY ERROR SHOULD CAUSE A CACHE
1855                                  ;MISS.
1856                                  ;
1857                                  ;BGNTST
1858                                  ;SAVE CONTENTS OF VECTOR 114
1859                                  ;LET VECTOR 114 POINT TO ABORT ROUTINE
1860                                  ;CLEAR MSER
```

```
                                ;SET WRITE WRONG TAG PARITY BIT<10>
                                ;WRITE TEST LOCATION
                                ;CLEAR CCR BIT<10> AND SET ABORT DISABLE BIT<0>
                                ;READ TEST LOCATION
                                ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ELSE
                                ;.      .       ERROR IN CACHE SYSTEM
                                ;ENDIF
                                ;RESTORE VECTOR 114
                                ;EXIT   TST
                                ;
                                ;TAG PARITY ABORT ROUTINE:
                                ;                               ERROR IN CACHE SYSTEM
                                ;                               RETURN
                                ;ENDTST
                                ;;*****************************************************************
004716                          TST13:
004716  005267  174062                  INC     $TESTN                  ;INCREMENT TEST NUMBER
                                ;********************* NOTE **************************
                                ;   DON'T SINGLE STEP THIS CODE USING THE ODT P COMMAND!!!
                                ;   PARITY ERRORS WILL OCCUR.
                                ;*****************************************************
004722  013767  000114  174100          MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
004730  012737  005062  000114          MOV     #TAPAB0,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
004736  005067  173002                  CLR     MSER                    ;CLEAR MSER
004742  012767  002001  172776          MOV     #2001,  CCR             ;SET WRITE WRONG TAG PARITY & NO PARITY INTERRUP
004750  005037  001122                  CLR     @#TSTLOC                ;WRITE LOCATION WITH BAD TAG PARITY
004754  012767  000001  172764          MOV     #BIT00, CCR             ;CLEAR BIT 10 AND SET NO PARITY INTERRUPTS
004762  005737  001122                  TST     @#TSTLOC                ;READ TEST LOCATION
004766  032767  000010  172756          BIT     #BIT03, HITMIS          ;IF BIT 3 SET IN HIT/MISS REGISTER
004774  001414                          BEQ     2$                      ;THEN
004776  012737  004722  001120          MOV     #TST13+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
005004  004767  174630                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
005010  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
005012  000041                          .WORD   41                      ;UNIQUE ERROR NUMBER
005014  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                        ;CACHE SYSTEM ERROR
005016  000403                          BR      2$                      ;ELSE
005020                          1$:
005020  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
005022  000042                          .WORD   42                      ;UNIQUE ERROR NUMBER
005024  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                        ;CACHE SYSTEM ERROR
005026  012767  000400  172712  2$:     MOV     #BIT08,CCR              ;FLUSH CACHE, CLEAR CCR BEFORE EXIT
005034  016737  173770  000114          MOV     SLOC00, @#114           ;RESTORE CONTENTS OF VECTOR 114
005042  005037  177744                  CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
005046  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
005052  005037  001116                  CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
005056  000167  000010                  JMP     TST14                   ;JUMP OVER DATA TABLES OR SUBROUTINES
                                                                        ;TO NEXT TEST
005062                          TAPAB0:
005062  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
005064  000043                          .WORD   43                      ;UNIQUE ERROR NUMBER
005066  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
```

```
1917
1918  005070  000002                          RTI                                     ;CACHE SYSTEM ERROR
1919
1920
1921  005072                          CTST15:
1922                                  ;;*************************************************************************
1923                                  ;*TEST 14        TEST PARITY INTERRUPT
1924                                  ;;*************************************************************************
1925                                  ;PARITY INTERRUPT TEST - THIS TEST WILL VERIFY THAT WHEN CCR<7,0> =
1926                                  ;0,0, A PARITY INTERRUPT OCCURS AFTER EXECUTION OF AN INSTRUCTION
1927                                  ;THAT HAS BEEN WRITTEN WITH WRONG PARITY.
1928                                  ;
1929                                  ;BGNTST
1930                                  ;SAVE CONTENTS OF 114
1931                                  ;SETUP VECTOR 114 TO POINT TO INTERRUPT ROUTINE
1932                                  ;CLEAR EXPECTING INTERRUPT FLAG
1933                                  ;WRITE TEST ADDRESS WITH BAD PARITY
1934                                  ;SET EXPECTING INTERRUPT FLAG
1935                                  ;READ TEST ADDRESS
1936                                  ;IF INTERRUPT FLAG NE 0 THEN
1937                                  ;.      ERROR IN CACHE SYSTEM
1938                                  ;ENDIF
1939                                  ;RESTORE CONTENTS OF VECTOR 114
1940                                  ;EXIT   TST
1941                                  ;
1942                                  ;INTERRUPT ROUTINE:     IF EXPECTING INTERRUPT FLAG NE 1 THEN
1943                                  ;                       .       ERROR IN CACHE SYSTEM
1944                                  ;                       ELSE
1945                                  ;                       .       CLEAR (EXPECTING) INTERRUPT FLAG
1946                                  ;                       ENDIF
1947                                  ;                       IF SAVED PC NE TO UPDATED PC THEN
1948                                  ;                       .       ERROR IN CACHE SYSTEM
1949                                  ;                       ENDIF
1950                                  ;                       IF MSER NE #340 THEN
1951                                  ;                       .       ERROR IN CACHE SYSTEM
1952                                  ;                       ENDIF
1953                                  ;                       RETURN
1954                                  ;
1955                                  ;ENDTST
1956                                  ;;*************************************************************************
1957  005072
1958  005072  005267  173706         TST14:  INC     $TESTN                  ;INCREMENT TEST NUMBER
1959  005076  004767  174310                 JSR     PC,INITMM               ;INIT MMU; PAGE ZERO REFERENCES WILL
1960                                                                          ;BYPASS CACHE.
1961  005102  012767  001600  165244         MOV     #1600,KIPAR6            ;SET PAR6 TO RELOCATE TO BANK 7
1962  005110  012737  140000  001122         MOV     #140000,@#TSTLOC        ;SET UP TEST LOCATION TO OBTAIN
1963                                                                          ;PHYSICAL ADDR VIA KIPAR6.
1964  005116  052767  000001  172446         BIS     #BIT00,SR0              ;TURN ON MMU
1965  005124  005001                         CLR     R1                      ;CLEAR EXPECTING INTERRUPT FLAG
1966  005126  052767  000100  172612         BIS     #BIT06, CCR             ;SET WRITE WRONG DATA PARITY
1967  005134  005077  173762                 CLR     @TSTLOC                 ;WRITE CACHE LOCATION 0 WITH BAD DATA PARITY
1968  005140  005067  172602                 CLR     CCR                     ;CLEAR WRITE WRONG DATA PARITY
1969  005144  005037  177744                 CLR     @#MSER                  ;INIT MSER
1970  005150  013767  000114  173652         MOV     @#114,SLOC00            ;SAVE CONTENTS OF VECTOR 114
1971  005156  012737  005244  000114         MOV     #INTERR,@#114           ;LET VECTOR POINT TO INTERRUPT ROUTINE.
1972  005164  005101                         COM     R1                      ;SET EXPECTING INTERRUPT FLAG
```

```
1973  005166  005777  173730          INTRPC: TST     @TSTLOC                 ;READ TEST LOCATION; SHOULD GET PARITY INTERRUPT
1974  005172  005701                          TST     R1                      ;RETURN HERE FROM INTERRUPT
1975                                                                          ;IF INTERRUPT FLAG NOT EQUAL ZERO
1976  005174  001403                          BEQ     1$                      ;THEN WE DIDN'T GET A PARITY INTERRUPT.
1977  005176  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1978  005200  000044                          .WORD   44                      ;UNIQUE ERROR NUMBER
1979  005202  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1980                                                                                  ;CACHE SYSTEM ERROR
1981  005204  016737  173620  000114  1$:     MOV     SLOC00, @#114           ;RESTORE VECTOR 114
1982  005212  005067  172354                  CLR     SR0                     ;TURN OFF MMU
1983  005216  012737  000400  177746          MOV     #BIT08,@#CCR            ;FLUSH CACHE, CLEAR CCR
1984  005224  005037  177744                  CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
1985  005230  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
1986  005234  005037  001116                  CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
1987  005240  000167  000052                  JMP     TST15                   ;JUMP OVER DATA TABLES OR SUBROUTINES
1988                                                                          ;TO NEXT TEST
1989
1990  005244  005701                  INTERR: TST     R1                      ;IF EXPECTING INTERRUPT FLAG NOT SET
1991  005246  001004                          BNE     1$                      ;THEN
1992  005250  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
1993  005252  000045                          .WORD   45                      ;UNIQUE ERROR NUMBER
1994  005254  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
1995                                                                                  ;CACHE SYSTEM ERROR
1996  005256  000401                          BR      2$                      ;ELSE
1997  005260  005001                  1$:     CLR     R1                      ;CLEAR (EXPECTING) INTERRUPT FLAG
1998  005262  021627  005172          2$:     CMP     @SP,    #INTRPC+4       ;IF SAVED PC NOT EQUAL TO UPDATED PC
1999  005266  001403                          BEQ     4$                      ;THEN
2000  005270  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2001  005272  000046                          .WORD   46                      ;UNIQUE ERROR NUMBER
2002  005274  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2003                                                                                  ;CACHE SYSTEM ERROR
2004  005276  022767  000340  172440  4$:     CMP     #340,   MSER            ;IF MSER NOT EQUAL TO EXPECTED VALUE
2005  005304  001403                          BEQ     5$                      ;THEN
2006  005306  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2007  005310  000047                          .WORD   47                      ;UNIQUE ERROR NUMBER
2008  005312  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2009                                                                                  ;CACHE SYSTEM ERROR
2010  005314  000002                  5$:     RTI                             ;RETURN
2011
2012
2013  005316                          CTST17:
2014                                  ;;**************************************************************************
2015                                  ;*TEST 15       TEST THAT PARITY ERRORS BLOCKED BY NXM ABORT
2016                                  ;;**************************************************************************
2017                                  ;CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT - THIS TEST WILL
2018                                  ;VERIFY THAT IF A PARITY ERROR OCCURS ON THE SAME ADDRESS REFERENCE AS A
2019                                  ;NON-EXISTENT MEMORY ERROR THAT THE CACHE DATA PATH GATE ARRAY BLOCKS THE
2020                                  ;PARITY ERROR TO THE J-11 CHIP SET. THIS WILL BE DONE BY USING THE DIAGNOSTIC
2021                                  ;BIT TO CAUSE A PARITY ERROR IN A CACHE REFERENCE THAT DOES NOT HAVE A
2022                                  ;CORRESPONDING ADDRESS IN MAIN MEMORY. THE ADDRESS WILL THEN BE READ CAUSING
2023                                  ;BOTH A CACHE PARITY ERROR AND A NON-EXISTENT MEMORY ERROR. TO AVOID HAVING
2024                                  ;TO SIZE THE ENTIRE MEMORY TO FIND A NON-EXISTENT MEMORY ADDRESS THIS TEST
2025                                  ;WILL USE THE LARGEST NON-I/O ADDRESS (17757776). THE TEST WILL FIRST READ
2026                                  ;THIS ADDRESS AND IF A NXM TRAP OCCURS THE TEST WILL BE DONE. IF THE ACCESS
2027                                  ;TO THIS ADDRESS DOES NOT TRAP THEN THE TEST WILL BE SKIPPED.
2028                                  ;
```

```
                                ;BGNTST
                                ;SAVE CONTENTS OF VECTOR 4
                                ;SAVE CONTENTS OF VECTOR 114
                                ;LET VECTOR 4 POINT TO CONTINUE TESTING (A:)
                                ;LET PAR6  #177400
                                ;ACCESS ADDRESS 157776 (PHYSICAL 17757776)
                                ;IF NO TRAP THEN
                                ;.      GOTO ENDTST
                                ;ENDIF
                                ;A:
                                ;SET DIAGNOSTIC AND WRITE WRONG PARITY BITS IN CCR
                                ;WRITE ADDRESS 157776
                                ;CLEAR CCR
                                ;SET PARITY ERROR ABORT BIT IN CCR
                                ;LET VECTOR 4 POINT TO CONTINUE TESTING (B:)
                                ;LET VECTOR 114 POINT TO NXM-PARITY ERROR ROUTINE
                                ;READ ADDRESS 157776
                                ;IF NO TRAP THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ENDIF
                                ;B:
                                ;CLEAR CCR
                                ;RESTORE CONTENTS OF VECTOR 114
                                ;RESTORE CONTENTS OF VECTOR 4
                                ;EXIT   TST
                                ;
                                ;
                                ;NXM-PARITY ERROR ROUTINE:      RESET STACK AFTER TRAP
                                ;                               ERROR IN CACHE SYSTEM
                                ;                               GOTO B:
                                ;ENDTST
                                ;;*****************************************************************************
005316                          TST15:
005316  005267  173462                  INC     $TESTN                  ;INCREMENT TEST NUMBER
005322  013704  000004                  MOV     @#4,R4                  ;SAVE CONTENTS OF VECTOR 4
005326  013767  000114  173474          MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
005334  012737  005372  000004          MOV     #1$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
005342  012767  177400  165004          MOV     #177400,KIPAR6          ;LET PAR6 = OFFSET TO HIGHEST MEMORY
005350  012767  000020  165140          MOV     #20,SR3                 ;ENABLE 22 BIT ADDRESSING.
005356  052737  000001  177572          BIS     #BIT00,@#SR0            ;ENABLE MMU
005364  005737  157776                  TST     @#157776                ;ACCESS ADDRESS 17757776
005370  000435                          BR      3$                      ;IF IT DOESN'T TRAP THEN SKIP THIS TEST
005372  062706  000004          1$:     ADD     #4,     SP              ;RESET STACK AFTER TRAP
005376  012767  000102  172342          MOV     #102,   CCR             ;SET DIAG. AND WRITE WRONG DATA PARITY BITS
005404  005037  157776                  CLR     @#157776                ;WRITE TO ADDRESS 17757776
005410  012767  000200  172330          MOV     #200,   CCR             ;CLEAR CCR AND SET PARITY ABORT BIT
005416  012737  005460  000004          MOV     #2$,    @#4             ;LET VECTOR 4 POINT TO CONTINUE TESTING
005424  012737  005446  000114          MOV     #100$,@#114             ;LET VECTOR 114 POINT TO ERROR ROUTINE
005432  005737  157776                  TST     @#157776                ;READ ADDRESS 17757776 (SHOULD TRAP)
005436  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
005440  000050                          .WORD   50                      ;UNIQUE ERROR NUMBER
005442  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                ;CACHE SYSTEM ERROR
005444  000407                          BR      3$                      ;GO EXIT TEST
005446  005067  172120          100$:   CLR     SR0                     ;TURN OFF MMU BEFORE ERROR
005452  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
```

```
2085  005454  000051                      .WORD   51                      ;UNIQUE ERROR NUMBER
2086  005456  001325                      .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2087                                                                        ;CACHE SYSTEM ERROR
2088  005460  062706  000004        2$:   ADD     #4,     SP              ;RESET STACK AFTER TRAP
2089  005464  005067  172256        3$:   CLR     CCR                     ;CLEAR CCR FOR EXIT
2090  005470  010437  000004              MOV     R4,     @#4             ;RESTORE VECTOR 4
2091  005474  016737  173330  000114      MOV     SLOC00, @#114           ;RESTORE VECTOR 114
2092  005502  005067  172064              CLR     SR0                     ;TURN OFF MMU BEFORE EXITING.
2093  005506  005037  177744              CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
2094  005512  005037  177766              CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
2095  005516  005037  001116              CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
2096  005522  000167  000000              JMP     TST16                   ;JUMP OVER DATA TABLES OR SUBROUTINES
2097                                                                        ;TO NEXT TEST
2098
2099
2100
2101  005526                      CTST20:
2102                              ;;*******************************************************************
2103                              ;*TEST 16       TEST MULTIPROCESSING INSTRUCTIONS
2104                              ;;*******************************************************************
2105                              ;MULTIPROCCESSING INSTRUCTION TESTS - THIS TEST WILL VERIFY THAT THE MULTI-
2106                              ;PROCCESSING INSTRUCTIONS DO A BYPASS OF THE CACHE. THIS TEST WILL NOT VERIFY
2107                              ;THE REST OF THE FUNCTIONALITY OF THESE INSTRUCTIONS BECAUSE THAT WILL ALREADY
2108                              ;HAVE BEEN CHECKED IN THE BASE INSTRUCTION TESTS.
2109                              ;THE TEST FOR TSTSET AND WRTLCK WILL FIRST ALLOCATE AN ADDRESS IN CACHE WITH
2110                              ;KNOWN DATA, THEN FORCE CACHE MISS WILL BE ENABLED AND THE SAME ADDRESS WILL
2111                              ;BE WRITTEN WITH WITH DIFFERENT DATA. THIS WILL SET UP CACHE WITH DATA 1 AND
2112                              ;IT'S ASSOCIATED MEMORY LOCATION WITH DATA 2. FORCE CACHE MISS WILL THEN BE
2113                              ;DISABLED AND THE TSTSET/WRTLCK INSTRUCTION WILL BE EXECUTED. THIS INSTRUCTION
2114                              ;WILL CAUSE CACHE TO BE BYPASSED AND AN ACCESS TO MAIN MEMORY WILL BE MADE. IN
2115                              ;ADDITION THIS INSTRUCTION WILL CAUSE CACHE TO BE RE-ALLOCATED WITH THE CORRECT;DATA.
2116                              ;THE TEST FOR ASRB INSTRUCTION DIFFERS SLIGHTLY. FIRST THE CACHE WILL BE
2117                              ;ALLOCATED, THEN AN ASRB INSTRUCTION WILL BE DONE AT THAT ADDRESS.
2118                              ;A HIT SHOULD BE RECORDED ON THE ACCESS. NEXT, THE ADDRESS WILL BE READ AND
2119                              ;A MISS SHOULD BE RECORDED BECAUSE THE FORCED BYPASS ON THE ASRB INSTRUCTION
2120                              ;SHOULD HAVE INVALIDATED THE CACHE ENTRY, AND BECAUSE IT'S A BYTE WRITE; CACHE
2121                              ;WILL NOT BE RE-ALLOCATED.
2122                              ;                       *****NOTE*****
2123                              ;IF THE TSTSET OR WRTLCK ISTRUCTIONS ARE ASSEMBLED SO THAT BITS
2124                              ;5-0 OF THE ADDRESS OF THE INSTRUCTION FOLLOWING THE TSTSET/WRTLCK INSTRUCTION
2125                              ;ARE EQUAL TO BITS 5-0 OF THE TSTLOC ADDRESS THE TEST WILL FAIL. THIS IS CAUSED
2126                              ;BY THE NEED OF AN ADDITIONAL FETCH WHEN THE PRE-FETCH BUFFER IS INVALIDATED
2127                              ;WHEN BITS 5-0 OF AN ADDRESS WRITTEN TO ARE EQUAL TO BITS 5-0 OF THE ADDRESS
2128                              ;IN THE PREFETCH BUFFER. THE ADDITIONAL FETCH CAUSES THE REQUIRED DATA TO BE
2129                              ;SHIFTED OUT OF THE RANGE OF THE HIT/MISS REGISTER. PADDING THE FRONT OF THE
2130                              ;TEST WITH NOPS MAY BE REQUIRED TO REMEDY THIS PROBLEM.
2131                              ;BGNTST
2132                              ;READ TEST LOCATION TO ALLOCATE CACHE
2133                              ;DO TSTSET INSTRUCTION
2134                              ;IF HIT/MISS REGISTER BIT 3 NOT SET THEN
2135                              ;.      ERROR IN CACHE SYSTEM
2136                              ;ENDIF
2137                              ;READ TEST LOCATION
2138                              ;IF HIT/MISS REGISTER BIT 3 SET THEN
2139                              ;.      ERROR IN CACHE SYSTEM
2140                              ;ENDIF
```

```
                                ;READ TEST LOCATION TO ALLOCATE CACHE
                                ;DO WRTLCK INSTRUCTION
                                ;IF HIT/MISS REGISTER BIT 3 NOT SET THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ENDIF
                                ;READ TEST LOCATION
                                ;IF HIT/MISS REGISTER BIT 3 SET THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ENDIF
                                ;READ TEST LOCATION TO ALLOCATE CACHE
                                ;DO ASRB INSTRUCTION
                                ;IF HIT/MISS REGISTER BIT 3 NOT SET THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ENDIF
                                ;READ TEST LOCATION
                                ;IF HIT/MISS REGISTER BIT 3 SET THEN
                                ;.      ERROR IN CACHE SYSTEM
                                ;ENDIF
                                ;ENDTST
                                ;;*********************************************************************
005526                          TST16:
005526  005267  173252                  INC     $TESTN                  ;INCREMENT TEST NUMBER
005532  005067  173364                  CLR     TSTLOC                  ;WRITE TEST LOCATION TO ALLOCATE CACHE
005536  012767  000004  172202          MOV     #BIT02,CCR              ;SET CACHE TO FORCE MISS
005544  012737  177777  001122          MOV     #177777,@#TSTLOC        ;WRITE TO MEMORY BUT NOT TO CACHE.
                                                                        ;CACHE=000000; TSTLOC=177777
005552  005037  177746                  CLR     @#CCR                   ;CLEAR FORCE MISS
                                        ;       TSTSET  TSTLOC                  ;DO TSTSET INSTRUCTION
005556  007237                          .WORD   7237                    ;THESE NEXT TWO LOCATIONS ARE THE TSTSET
005560  001122                          .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
005562  001415                          BEQ     100$                    ;IF = 0 THEN CACHE WAS NOT BYPASSED
005564  032767  000040  172160          BIT     #BIT05,HITMIS           ;IF HIT/MISS REGISTER BIT 5 IS SET
005572  001014                          BNE     1$                      ;THEN GO TO NEXT TEST.
005574  012737  005532  001120          MOV     #TST16+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
005602  004767  174032                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
005606  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
005610  000052                          .WORD   52                      ;UNIQUE ERROR NUMBER
005612  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                ;CACHE SYSTEM ERROR
                                                                        ;ERROR! BIT 5 IN HIT MISS REG WAS NOT SET.
005614  000403                          BR      1$                      ;RECOVER FROM ERROR
005616                          100$:
005616  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
005620  000053                          .WORD   53                      ;UNIQUE ERROR NUMBER
005622  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
                                                                                ;CACHE SYSTEM ERROR
                                                                        ;ERROR! CACHE WAS NOT BYPASSED BY TSTSET
005624  022767  177777  173270  1$:     CMP     #177777,TSTLOC          ;READ TEST LOCATION (ALSO ALLOCATES CACHE FOR WR
005632  001015                          BNE     200$                    ;CACHE SHOULD BE RE-ALLOCATED WITH
                                                                        ;177777 AS A RESULT OF TSTSET INSTRUCTION.
005634  032767  000020  172110          BIT     #BIT04, HITMIS          ;IF HIT/MISS REGISTER BIT 4 SET (SHOULD BE)
005642  001014                          BNE     2$                      ;THEN
005644  012737  005532  001120          MOV     #TST16+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
005652  004767  173762                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
005656  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
005660  000054                          .WORD   54                      ;UNIQUE ERROR NUMBER
```

```
2197  005662  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2198                                                                                ;CACHE SYSTEM ERROR
2199  005664  000403                        BR      2$                      ;RECOVER FROM ERROR
2200  005666                          200$:
2201  005666  104000                        ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2202  005670  000055                        .WORD   55                      ;UNIQUE ERROR NUMBER
2203  005672  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2204                                                                                ;CACHE SYSTEM ERROR
2205                                                                        ;ERROR! CACHE WAS NOT RE-ALLOCATED
2206  005674  012767  000004  172044  2$:   MOV     #BIT02,CCR              ;SET FORCE CACHE MISS.
2207  005702  005067  173214                CLR     TSTLOC                  ;CACHE=177777; TSTLOC=000000
2208  005706  005000                        CLR     R0                      ;CLR R0
2209  005710  005067  172032                CLR     CCR                     ;CLEAR FORCE CACHE MISS.
2210                                    ;   WRTLCK   TSTLOC                  ;DO WRTLCK INSTRUCTION
2211  005714  007337                        .WORD   7337                    ;THESE NEXT TWO WORDS ARE THE WRTLCK
2212  005716  001122                        .WORD   TSTLOC                  ;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
2213  005720  001015                        BNE     300$                    ;
2214  005722  032767  000040  172022        BIT     #BIT05, HITMIS          ;IF HIT/MISS REGISTER BIT 5 NOT SET
2215  005730  001014                        BNE     3$                      ;THEN
2216  005732  012737  005532  001120        MOV     #TST16+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
2217  005740  004767  173674                JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
2218  005744  104000                        ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2219  005746  000056                        .WORD   56                      ;UNIQUE ERROR NUMBER
2220  005750  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2221                                                                                ;CACHE SYSTEM ERROR
2222  005752  000403                        BR      3$                      ;RECOVER FROM ERROR
2223  005754                          300$:
2224  005754  104000                        ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2225  005756  000057                        .WORD   57                      ;UNIQUE ERROR NUMBER
2226  005760  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2227                                                                                ;CACHE SYSTEM ERROR
2228                                                                        ;ERROR! CACHE WAS NOT BYPASSED BY WRTLCK.
2229  005762  005767  173134          3$:   TST     TSTLOC                  ;READ TEST LOCATION
2230  005766  001015                        BNE     400$                    ;
2231  005770  032767  000020  171754        BIT     #BIT04, HITMIS          ;IF HIT/MISS REGISTER BIT 4 SET
2232  005776  001014                        BNE     4$                      ;THEN
2233  006000  012737  005532  001120        MOV     #TST16+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
2234  006006  004767  173626                JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
2235  006012  104000                        ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2236  006014  000060                        .WORD   60                      ;UNIQUE ERROR NUMBER
2237  006016  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2238                                                                                ;CACHE SYSTEM ERROR
2239  006020  000403                        BR      4$                      ;RECOVER FROM ERROR
2240  006022                          400$:
2241  006022  104000                        ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2242  006024  000061                        .WORD   61                      ;UNIQUE ERROR NUMBER
2243  006026  001325                        .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2244                                                                                ;CACHE SYSTEM ERROR
2245                                                                        ;ERROR! CACHE WAS NOT RE-ALLOCATED
2246  006030  005767  173066          4$:   TST     TSTLOC                  ;READ TEST LOCATION TO ALLOCATE CACHE
2247  006034  000240                        NOP                             ;NOP PAD FOR ASRB HIT/MISS CALC
2248  006036  106267  173060                ASRB    TSTLOC                  ;DO ASRB INSTRUCTION
2249  006042  032767  000020  171702        BIT     #BIT04, HITMIS          ;IF HIT/MISS REGISTER BIT 4 NOT SET
2250  006050  001010                        BNE     5$                      ;THEN
2251  006052  012737  005532  001120        MOV     #TST16+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
2252  006060  004767  173554                JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
```

```
2253  006064  104000                        ERROR                   ;ALL ERRORS TO TRAP TO EMT VECTOR
2254  006066  000062                        .WORD   62              ;UNIQUE ERROR NUMBER
2255  006070  001325                        .WORD   ERRMSG          ;ADDRESS OF ERROR MESSAGE
2256                                                                  ;CACHE SYSTEM ERROR
2257  006072  005767  173024        5$:     TST     TSTLOC          ;READ TEST LOCATION
2258  006076  032767  000010  171646        BIT     #BIT03, HITMIS  ;IF HIT/MISS REGISTER BIT 3 SET
2259  006104  001410                        BEQ     6$              ;THEN
2260  006106  012737  005532  001120        MOV     #TST16+4,@#SOFTRE ;MOVE RETRY ADDRESS TO SOFT RETURN
2261  006114  004767  173520                JSR     PC,APTSFT       ;GO TO SOFT ERROR SUBROUTINE
2262  006120  104000                        ERROR                   ;ALL ERRORS TO TRAP TO EMT VECTOR
2263  006122  000063                        .WORD   63              ;UNIQUE ERROR NUMBER
2264  006124  001325                        .WORD   ERRMSG          ;ADDRESS OF ERROR MESSAGE
2265                                                                  ;CACHE SYSTEM ERROR
2266  006126                        6$:
2267  006126  005037  177744                CLR     @#MSER          ;CLEAR THE MEM SYS ERROR REG
2268  006132  005037  177766                CLR     @#CPEREG        ;CLEAR THE CPU ERROR REGISTER
2269  006136  005037  001116                CLR     @#SOFTER        ;CLEAR THE SOFT ERROR COUNT
2270  006142  000167  000000                JMP     TST17           ;JUMP OVER DATA TABLES OR SUBROUTINES
2271                                                                  ;TO NEXT TEST
2272
2273
2274
2275  006146                        CTST21:
2276                                ;;*****************************************************************
2277                                ;*TEST 17       TEST CACHE DATA RAMS
2278                                ;;*****************************************************************
2279                                ;DATA STORE RAM TESTS - THERE ARE TWO TESTS FOR THE DATA STORE RAM.
2280                                ;THE FIRST TEST WILL BE A NO DUAL ADDRESSING TEST AND THE SECOND
2281                                ;TEST WILL BE A DATA RELIABILITY TEST. THE NO DUAL ADDRESSING TEST
2282                                ;WILL FIRST WRITE EACH WORD OF THE CACHE RAM WITH ITS WORD ADDRESS
2283                                ;AND VERIFY THE CONTENTS WITH A READ OF EACH ADDRESS. THEN EACH
2284                                ;BYTE WILL BE WRITTEN WITH ITS ADDRESS AND CHECKED. THE DATA
2285                                ;RELIABILITY TEST THAT WILL BE USED IS A TEST CALLED MOVING INVERSIONS.
2286                                ;
2287                                ;BGNTST 1
2288                                ;SETUP MMU REGISTERS TO HAVE BYPASS ON KERNAL SPACE AND NO
2289                                ;       BYPASS ON USER SPACE
2290                                ;LET PS EQUAL KERNAL FOR CURRENT MODE AND USER FOR PREVIOUS
2291                                ;       MODE
2292                                ;ENABLE MMU
2293                                ;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
2294                                ;CLEAR DATA TO BE WRITTEN
2295                                ;SET DIAGNOSTIC BIT (BIT<1>) IN CCR
2296                                ;DO UNTIL DATA TO BE WRITTEN EQUALS 20000(8)
2297                                ;.      WRITE DATA TO ADDRESS
2298                                ;.      ADD 2 TO ADDRESS
2299                                ;.      ADD 2 TO DATA TO BE WRITTEN
2300                                ;ENDDO
2301                                ;SAVE CONTENTS OF VECTOR 114
2302                                ;LET VECTOR 114 POINT TO DATA STORE PARITY ABORT ROUTINE
2303                                ;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
2304                                ;CLEAR EXPECTED DATA
2305                                ;DO UNTIL EXPECTED DATA EQUALS 20000(8)
2306                                ;.      READ ADDRESS
2307                                ;.      IF RECEIVED DATA NE EXPECTED DATA THEN
2308                                ;.      ERROR IN CACHE SYSTEM
```

```
                                        ;.      ENDIF
                                        ;.      ADD 2 TO EXPECTED DATA
                                        ;.      ADD 2 TO ADDRESS
                                        ;ENDDO
                                        ;GET FIRST ADDRESS OF 8K BYTE DATA BUFFER
                                        ;CLEAR DATA TO BE WRITTEN
                                        ;DO UNTIL ALL BYTES CHECKED
                                        ;.      WRITE DATA TO ADDRESS
                                        ;.      ADD 1 TO ADDRESS
                                        ;.      ADD 1 TO DATA TO BE WRITTEN
                                        ;ENDDO
                                        ;GET FIRST ADDRESS OF 8K BYTE DATA BUFFER
                                        ;CLEAR EXPECTED DATA
                                        ;DO UNTIL EXPECTED DATA EQUALS 20000(8)
                                        ;.      READ ADDRESS
                                        ;.      IF RECEIVED DATA NE EXPECTED DATA THEN
                                        ;.      ERROR IN CACHE SYSTEM
                                        ;.      ENDIF
                                        ;.      ADD 1 TO EXPECTED DATA
                                        ;.      ADD 1 TO ADDRESS
                                        ;ENDDO
                                        ;RESTORE VECTOR 114
                                        ;ENDTST
                                        ;
                                        ;;*********************************************************************
006146                                  TST17:
006146  005267  172632                          INC     $TESTN                  ;INCREMENT TEST NUMBER
006152  004767  173234                          JSR     PC,     INITMM          ;SETUP MMU REGISTERS/FORCE CACHE BYPASS PAGES 0-
006156  012767  001600  164170                  MOV     #1600,  KIPAR6          ;LET PAR6 MAP TO ADDRESS 160000
006164  042767  000004  164324                  BIC     #BIT02,SR3              ;DISABLE KERNEL D SPACE.
006172  052737  000001  177572                  BIS     #BIT00,@#SR0            ;TURN ON MEMORY MANAGEMENT
006200  012702  140000                          MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
006204  005001                                  CLR     R1                      ;INIT DATA TO BE WRITTEN
006206  052767  000002  171532                  BIS     #BIT01,CCR              ;SET DIAGNOSTIC BIT
006214  020127  020000          1$:             CMP     R1,     #20000          ;DO UNTIL ALL ADDRESSES WRITTEN
006220  001404                                  BEQ     2$                      ;IF DONE GO CHECK DATA
006222  010122                                  MOV     R1,     (R2)+           ;WRITE DATA TO ADDRESS
006224  062701  000002                          ADD     #2,     R1              ;UPDATE DATA BY 2 (WORD BOUNDARY)
006230  000771                                  BR      1$                      ;ENDDO
006232  013767  000114  172570  2$:             MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
006240  012737  001630  000114                  MOV     #RAMPAR,@#114           ;LET VECTOR 114 POINT TO ABORT ROUTINE
006246  042767  000002  171472                  BIC     #BIT01,CCR              ;CLEAR DIAGNOSTIC BIT. ALL FURTHER
                                                                                ;REFERENCES TO PAGE 6 SHOULD BE CACHED.
006254  012702  140000                          MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
006260  005001                                  CLR     R1                      ;INIT DATA TO BE CHECKED
006262  020127  020000          3$:             CMP     R1,     #20000          ;DO UNTIL ALL ADDRESSES CHECKED
006266  001430                                  BEQ     5$                      ;IF DONE GO DO BYTES
006270  005712                                  TST     (R2)                    ;CHECK THAT CACHE WAS
006272  032737  000010  177752                  BIT     #BIT03,@#HITMIS         ;ALLOCATED
006300  001010                                  BNE     4$                      ;SHOULD HAVE HIT
006302  012737  006152  001120                  MOV     #TST17+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
006310  004767  173324                          JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
                                                                                        ;CACHE SYSTEM ERROR
                                                                                        ;EXPECTED A HIT; GOT A MISS
006314  104000                          ERROR                                   ;ALL ERRORS TO TRAP TO EMT VECTOR
006316  000064                          .WORD   64                              ;UNIQUE ERROR NUMBER
```

```
2365  006320  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2366  006322  012267  172510          4$:     MOV     (R2)+,  RECDAT          ;READ TEST LOCATION
2367  006326  020167  172504                  CMP     R1,     RECDAT          ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
2368  006332  001403                          BEQ     100$                    ;THEN
2369  006334  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2370  006336  000065                          .WORD   65                      ;UNIQUE ERROR NUMBER
2371  006340  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2372                                                                                  ;CACHE SYSTEM ERROR
2373  006342  062701  000002          100$:   ADD     #2,R1                   ;UPDATE EXPECTED DATA BY 2
2374  006346  000745                          BR      3$                      ;ENDDO
2375
2376                                          ;***NOW CHECK BYTES****
2377                                          ;SINCE CACHE IS ALREADY ALLOCATED; WRITE BYTES WILL HIT
2378                                          ;AND WRITE THROUGH CACHE TO MEMORY.
2379
2380  006350  052737  000002  177746  5$:     BIS     #BIT01,@#CCR            ;SET DIAGNOSTIC MODE IN CCR
2381  006356  012702  140000                  MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
2382  006362  005001                          CLR     R1                      ;INIT DATA TO BE WRITTEN
2383  006364  020127  020000          6$:     CMP     R1,     #20000          ;DO UNTIL ALL ADDRESSES WRITTEN
2384  006370  001404                          BEQ     7$                      ;IF DONE GO CHECK DATA
2385  006372  110122                          MOVB    R1,     (R2)+           ;WRITE DATA TO ADDRESS
2386  006374  062701  000001                  ADD     #1,     R1              ;UPDATE DATA BY 1 (BYTE BOUNDARY)
2387  006400  000771                          BR      6$                      ;ENDDO
2388  006402  012702  140000          7$:     MOV     #140000,R2              ;GET FIRST ADDRESS OF 4K WORD BUFFER
2389  006406  005001                          CLR     R1                      ;INIT DATA TO BE CHECKED
2390                                                                          ;DO UNTIL ALL ADDRESSES CHECKED
2391  006410  105712                  8$:     TSTB    (R2)                    ;CHECK THAT CACHE WAS ALLOCATED
2392  006412  032737  000010  177752          BIT     #BIT03,@#HITMIS         ;SHOULD BE
2393  006420  001010                          BNE     101$                    ;       A HIT
2394  006422  012737  006152  001120          MOV     #TST17+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
2395  006430  004767  173204                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
2396                                                                                  ;CACHE SYSTEM ERROR
2397                                                                                  ;EXPECTED A HIT; GOT A MISS
2398  006434  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2399  006436  000066                          .WORD   66                      ;UNIQUE ERROR NUMBER
2400  006440  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2401  006442  112267  172370          101$:   MOVB    (R2)+,  RECDAT          ;READ TEST LOCATION
2402  006446  120167  172364                  CMPB    R1,     RECDAT          ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
2403  006452  001410                          BEQ     9$                      ;THEN
2404  006454  012737  006152  001120          MOV     #TST17+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
2405  006462  004767  173152                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
2406  006466  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2407  006470  000067                          .WORD   67                      ;UNIQUE ERROR NUMBER
2408  006472  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2409                                                                                  ;CACHE SYSTEM ERROR
2410  006474  005201                  9$:     INC     R1                      ;UPDATE EXPECTED DATA BY 1
2411  006476  020127  020000                  CMP     R1,#20000               ;IF ALL ADDRESS HAVE NOT BEEN TESTED
2412  006502  001342                          BNE     8$                      ;THEN GET NEXT ADDRESS
2413                                                                          ;ELSE EXIT TEST
2414  006504  005067  171236          10$:    CLR     CCR                     ;CLEAR THE CCR
2415  006510  005067  171056                  CLR     SR0                     ;TURN OFF MMU
2416  006514  016737  172310  000114          MOV     SLOC00, @#114           ;RESTORE VECTOR 114
2417  006522  005037  177744                  CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
2418  006526  005037  177766                  CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
2419  006532  005037  001116                  CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
2420  006536  000167  000000                  JMP     TST20                   ;JUMP OVER DATA TABLES OR SUBROUTINES
```

```
                                                                ;TO NEXT TEST

006542                                  CTST22:
                                        ;;**********************************************************************
                                        ;*TEST 20       CACHE DATA RAM DATA RELIABILITY TEST
                                        ;;**********************************************************************
                                        ;MOVING INVERSIONS TEST FOR DATA RAMS - THE TEST IS STARTED AFTER LOADING THE
                                        ;RAM STORE WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A
                                        ;1 IS SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE
                                        ;ADDRESS IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF
                                        ;THE WORD LEAVING THE ARRAY FILED WITH 1'S. THE WHOLE PROCESS IS REPEATED
                                        ;PLUGGING IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESSING IN THE DOWNWARD
                                        ;DIRECTION. FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE
                                        ;LSB. TO SAVE TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING
                                        ;ONLY A SINGLE BIT AT A TIME EVERY FOURTH BIT WILL BE DONE CONCURRENTLY.
                                        ;
                                        ;BGNTST
                                        ;SETUP AND ENABLE MMU
                                        ;SETUP CCR TO ABORT PARITY ERRORS
                                        ;CLEAR CACHE
                                        ;LET FWDSEQ = #1
                                        ;LET ADDLSB = #1
                                        ;DO UNTIL ADDLSB EQ #0
                                        ;.      LET CURDAT = 0
                                        ;.      LET RITEDA = #1
                                        ;.      LET NEWDAT = #1
                                        ;.      IF FWDSEQ = #1 THEN
                                        ;.      .       LET FSTADD EQUAL FIRST ADDRESS OF 8K BYTE BUFFER
                                        ;.      .       LET LASTAD EQUAL LAST ADDRESS OF 8K BYTE BUFFER
                                        ;.      ELSE
                                        ;.      .       LET FSTADD EQUAL LAST ADDRESS OF 8K BYTE BUFFER
                                        ;.      .       LET LASTAD EQUAL FIRST ADDRESS OF 8K BYTE BUFFER
                                        ;.      ENDIF
                                        ;.      LET CURADD = FSTADD
                                        ;.      LET DCOUNT = #0
                                        ;.      SAVE CONTENTS OF VECTOR 114
                                        ;.      LET VECTOR 114 POINT TO DATA STORE PARITY ABORT ROUTINE
                                        ;.      DO UNTIL DCOUNT EQ #10
                                        ;.      .       LET RECDAT = @CURADD
                                        ;.      .       IF RECDAT NE CURDAT THEN
                                        ;.      ERROR IN CACHE SYSTEM
                                        ;.      .       ENDIF
                                        ;.      .       IF DCOUNT GT #3 THEN
                                        ;.      .       .       LET @CURADD = @CURADD CLEARBY RITEDA
                                        ;.      .       ELSE
                                        ;.      .       .       LET @CURADD = @CURADD SETBY RITEDA
                                        ;.      .       ENDIF
                                        ;.      .       LET RECDAT = @CURADD
                                        ;.      .       IF RECDAT NE NEWDAT THEN
                                        ;.      ERROR IN CACHE SYSTEM
                                        ;.      .       ENDIF
                                        ;.      .       IF CURADD EQ LASTAD THEN
                                        ;.      .       .       IF RITEDA = #210 THEN
                                        ;.      .       .       .       IF DCOUNT NE #7 THEN
                                        ;.      .       .       .       .       LET CURDAT = #377
                                        ;.      .       .       .       .       LET RITEDA = #21
```

```
                                        ;.      .       .       .       .       LET NEWDAT = #356
                                        ;.      .       .       .       ENDIF
                                        ;.      .       .       ELSE
                                        ;.      .       .       .       LET CURDAT = NEWDAT
                                        ;.      .       .       .       ROTATE RITEDA
                                        ;.      .       .       .       IF DCOUNT GT #3 THEN
                                        ;.      .       .       .       .       LET NEWDAT = NEWDAT CLEAREDBY RITEDA
                                        ;.      .       .       .       ELSE
                                        ;.      .       .       .       .       LET NEWDAT = NEWDAT SETBY RITEDA
                                        ;.      .       .       .       ENDIF
                                        ;.      .       .       ENDIF
                                        ;.      .       .       INCREMENT DCOUNT
                                        ;.      .       .       LET CURADD = FSTADD
                                        ;.      .       ELSE
                                        ;.      .       .       IF FWDSEQ = #1 THEN
                                        ;.      .       .       .       LET CURADD = CURADD + ADDLSB
                                        ;.      .       .       .       IF CARRY THEN
                                        ;.      .       .       .       .       LET CURADD = CURADD + #1
                                        ;.      .       .       .       ENDIF
                                        ;.      .       .       ELSE
                                        ;.      .       .       .       LET CURADD = CURADD - ADDLSB
                                        ;.      .       .       .       IF CARRY THEN
                                        ;.      .       .       .       .       LET CURADD = LASTAD - ADDLSB
                                        ;.      .       .       .       .       LET CURADD = CURADD - #1
                                        ;.      .       .       .       ENDIF
                                        ;.      .       .       ENDIF
                                        ;.      .       ENDIF
                                        ;.      ENDDO
                                        ;.      IF FWDSEQ EQ #1 THEN
                                        ;.      .       LET FWDSEQ = #0
                                        ;.      ELSE
                                        ;.      .       ROTATE ADDLSB
                                        ;.      .       LET FWDSEQ = #1
                                        ;.      ENDIF
                                        ;ENDDO
                                        ;RESTORE VECTOR 114
                                        ;ENDTST
                                        ;R0=CURADD (CURRENT ADDRESS)
                                        ;R1=WORKING REGISTER
                                        ;R2=RECDAT (RECEIVED DATA)
                                        ;R3=RITEDA (DATA TO BE WRITTEN)
                                        ;R4=NEWDAT (EXPECTED DATA)
                                        ;R5=CURDAT (CURRENT DATA)
                                        ;;*************************************************************************
006542                                  TST20:
006542  005267  172236                          INC     $TESTN                  ;INCREMENT TEST NUMBER
006546  032777  001000  172272                  BIT     #BIT09,@SWR             ;ARE DATA RELIABILITY TESTS SELECTED?
006554  001002                                  BNE     100$                    ;IF BIT09 IN SWR = 1; DO RELIABILITY TEST
006556  000167  000454                          JMP     EXTST                   ;ELSE SKIP IT
006562  004767  172624                  100$:   JSR     PC,     INITMM          ;SETUP MEMORY MANAGEMENT
                                                                                ;PAGES 0 THRU 5 BYPASS CACHE
                                                                                ;PAGES 6 AND 7 DON'T BYPASS CACHE
006566  012767  000200  163560                  MOV     #200,KIPAR6             ;LET KIPAR6 RE-LOCATE TO 20000-37777
006574  052737  000001  177572                  BIS     #BIT00,@#SR0            ;TURN ON MMU
006602  052767  000200  171136                  BIS     #200,   CCR             ;ENABLE PARITY ABORTS
006610  012700  140000                          MOV     #140000,R0              ;---*
```

```
2533  006614  012701  010000            MOV     #10000, R1              ;   *
2534  006620  005020                1$: CLR     (R0)+                   ;   *CLEARING CACHE
2535  006622  077102                    SOB     R1,     1$              ;___*
2536  006624  012767  000001  172234    MOV     #1,     FWDSEQ          ;INIT UPWARD ADDRESSING INDICATOR
2537  006632  012767  000001  172230    MOV     #1,     ADDLSB          ;INIT LSB AND DO LOOP UNTIL SHIFTED OUT
2538  006640  013767  000114  172162    MOV     @#114,  SLOC00          ;SAVE CONTENTS OF VECTOR 114
2539  006646  012737  001630  000114    MOV     #RAMPAR,@#114           ;LET VECTOR POINT TO ABORT ROUTINE
2540  006654  005002                    CLR     R2                      ;INIT RECDAT.
2541  006656  005005                TSTLUP: CLR     R5                      ;INIT CURRENT DATA
2542  006660  012703  000021            MOV     #21,    R3              ;INIT DATA TO BE WRITTEN
2543  006664  012704  000021            MOV     #21,    R4              ;INIT EXPECTED DATA
2544  006670  005767  172172            TST     FWDSEQ                  ;IF ADDRESSING UPWARD
2545  006674  001407                    BEQ     1$                      ;THEN
2546  006676  012767  140000  172174    MOV     #140000,FSTADD          ;LET FIRST ADDRESS EQUAL LOWEST VALUE
2547  006704  012767  157777  172170    MOV     #157777,LSTADD          ;LET LAST ADDRESS EQUAL HIGHEST VALUE
2548  006712  000406                    BR      2$                      ;ELSE
2549  006714  012767  157777  172156  1$: MOV     #157777,FSTADD          ;LET FIRST ADDRESS EQUAL HIGHEST VALUE
2550  006722  012767  140000  172152    MOV     #140000,LSTADD          ;LET LAST ADDRESS EQUAL LOWEST VALUE
2551  006730  016700  172144        2$: MOV     FSTADD, R0              ;LET CURRENT ADDRESS EQUAL FIRST ADDRESS
2552  006734  005067  172114            CLR     DCOUNT                  ;INIT LOOP COUNTER
2553  006740  022767  000010  172106  BGNTLP: CMP     #8.,    DCOUNT          ;DO LOOP 8 TIMES (4 TIMES TO WRITE 1'S
2554  006746  001511                    BEQ     ENDTLP                  ;AND 4 TIMES TO WRITE BACK 0'S)
2555  006750  111002                    MOVB    (R0),R2                 ;FIRST READ LOCATION
2556  006752  120205                    CMPB    R2,     R5              ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
2557  006754  001403                    BEQ     1$                      ;DATA THEN
2558  006756  104000                    ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2559  006760  000070                    .WORD   70                      ;UNIQUE ERROR NUMBER
2560  006762  001325                    .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2561                                                                                ;CACHE SYSTEM ERROR
2562  006764  022767  000003  172062  1$: CMP     #3,     DCOUNT          ;IF LOOP COUNT GREATER THAN 3
2563  006772  002002                    BGE     2$                      ;THEN
2564  006774  140310                    BICB    R3,     (R0)            ;CLEAR TEST DATA BY WRITE DATA
2565  006776  000401                    BR      3$                      ;ELSE
2566  007000  150310                2$: BISB    R3,     (R0)            ;SET TEST DATA BY WRITE DATA
2567  007002  111002                3$: MOVB    (R0),R2                 ;DO READ AFTER WRITE
2568  007004  120204                    CMPB    R2,     R4              ;IF RECIEVED DATA NOT EQUAL TO EXPECTED
2569  007006  001403                    BEQ     4$                      ;DATA THEN
2570  007010  104000                    ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2571  007012  000071                    .WORD   71                      ;UNIQUE ERROR NUMBER
2572  007014  001325                    .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2573                                                                                ;CACHE SYSTEM ERROR
2574  007016  020067  172060        4$: CMP     R0,     LSTADD          ;IF CURRENT ADDRESS EQUALS LAST ADDRESS
2575  007022  001034                    BNE     8$                      ;THEN
2576  007024  022703  000210            CMP     #210,   R3              ;AND IF WRITE PATTERN EQUALS LAST
2577  007030  001013                    BNE     5$                      ;PATTERN THEN
2578  007032  022767  000007  172014    CMP     #7,     DCOUNT          ;AND TEST IS NOT ON LAST LOOP
2579  007040  001420                    BEQ     7$                      ;THEN
2580  007042  012705  000377            MOV     #377,   R5              ;SET UP TO WRITE 0'S
2581  007046  012703  000021            MOV     #21,    R3
2582  007052  012704  000356            MOV     #356,   R4
2583  007056  000411                    BR      7$
2584  007060  010405                5$: MOV     R4,     R5              ;ELSE
2585  007062  006103                    ROL     R3                      ;SET UP FOR NEXT DATA PATTERN
2586  007064  022767  000003  171762    CMP     #3,     DCOUNT          ;AND IF DOWNWARD ADDRESSING
2587  007072  002002                    BGE     6$                      ;THEN
2588  007074  040304                    BIC     R3,     R4              ;LET NEW DATA BE CLEARED BY WRITE DATA
```

```
2589  007076  000401                        BR      7$                      ;ELSE
2590  007100  050304                6$:     BIS     R3,     R4              ;LET NEWDAT BE SET BY WRITE DATA
2591  007102  005267  171746        7$:     INC     DCOUNT                  ;UPDATE LOOP COUNTER
2592  007106  016700  171766              MOV     FSTADD, R0              ;REINIT FIRST ADDRESS FOR THIS PASS
2593  007112  000426                        BR      10$                     ;ELSE
2594  007114  005767  171746        8$:     TST     FWDSEQ                  ;IF ADDRESSING UPWARD
2595  007120  001412                        BEQ     9$                      ;THEN
2596  007122  066700  171742              ADD     ADDLSB, R0              ;CALCULATE NEXT HIGHER ADDRESS
2597  007126  022700  160000              CMP     #160000,R0              ;IF CURRENT ADDRESS HAS BEEN INCREASED
2598  007132  003016                        BGT     10$                     ;ABOVE HIGHEST ADDRESS THEN
2599  007134  162700  020000              SUB     #20000, R0              ;ROLL ADDRESS BACK
2600  007140  062700  000001              ADD     #1,     R0
2601  007144  000411                        BR      10$                     ;ELSE
2602  007146  166700  171716        9$:     SUB     ADDLSB, R0              ;CALCULATE NEXT LOWER ADDRESS
2603  007152  022700  140000              CMP     #140000,R0              ;IF CURRENT ADDRESS HAS BEEN DECREASED
2604  007156  003404                        BLE     10$                     ;BELOW LOWEST ADDRESS THEN
2605  007160  062700  020000              ADD     #20000, R0              ;ROLL ADDRESS BACK
2606  007164  162700  000001              SUB     #1,     R0
2607  007170  000663                10$:    BR      BGNTLP                  ;ENDDO
2608  007172  005767  171670        ENDTLP: TST     FWDSEQ                  ;IF ADDRESSING UPWARD FINISHED
2609  007176  001404                        BEQ     1$                      ;THEN
2610  007200  005067  171662              CLR     FWDSEQ                  ;DO ADDRESSING DOWNWARD
2611  007204  000167  177446              JMP     TSTLUP
2612  007210  012767  000001  171650  1$:     MOV     #1,     FWDSEQ          ;SET ADDRESSING UPWARD INDICATOR
2613  007216  006167  171646              ROL     ADDLSB                  ;UPDATE LSB TO NEXT POSITION
2614  007222  022767  020000  171640  ENDLUP: CMP     #20000,ADDLSB           ;WE ONLY GO AS FAR AS 10000
2615  007230  001402                        BEQ     EXTST                   ;ENDDO
2616  007232  000167  177420              JMP     TSTLUP
2617  007236  016737  171566  000114  EXTST:  MOV     SLOCOO, @#114           ;RESTORE VECTOR 114
2618  007244  005067  170476              CLR     CCR                     ;INIT CCR FOR EXIT
2619  007250  005067  170316              CLR     SR0                     ;TURN OFF MMU
2620  007254  005037  177744              CLR     @#MSER                  ;CLEAR THE MEM SYS ERROR REG
2621  007260  005037  177766              CLR     @#CPEREG                ;CLEAR THE CPU ERROR REGISTER
2622  007264  005037  001116              CLR     @#SOFTER                ;CLEAR THE SOFT ERROR COUNT
2623  007270  000167  000000              JMP     TST21                   ;JUMP OVER DATA TABLES OR SUBROUTINES
2624                                                                        ;TO NEXT TEST
2625
2626  007274                        CTST23:
2627                                ;;**************************************************************************
2628                                ;*TEST 21        TAG RAM DUAL ADDRESS TEST
2629                                ;;**************************************************************************
2630                                ;TAG STORE RAM TESTS - THERE ARE TWO TESTS FOR THE TAG STORE RAMS. THE FIRST
2631                                ;TEST IS AN ADDRESSING TEST TO ENSURE NO DUAL ADDRESSING OF THE TAG STORE.
2632                                ;THE SECOND TEST IS A "GALLOPING ONES AND ZEROS" TEST THAT CHECKS THE DATA
2633                                ;RELIABILITY OF THE TAG RAMS.
2634                                ;
2635                                ;DUAL ADDRESSING TEST - THIS WILL BE DONE BY FIRST FLUSHING THE CACHE, THEN
2636                                ;THE FIRST 256(10) LOCATIONS (BLOCK 0) WILL BE WRITTEN WITH ADDRESSES THAT
2637                                ;WILL CAUSE A INCREMENTING PATTERN TO BE STORED IN THOSE LOCATIONS OF THE TAG
2638                                ;STORE RAM. NEXT THE ENTIRE 4K ADDRESS RANGE WILL BE READ. THE HIT/MISS
2639                                ;REGISTER SHOULD ONLY REPORT HITS ON THE ADDRESSES THAT WERE ALLOCATED. THIS
2640                                ;PROCESS WILL BE REPEATED FOR EACH BLOCK FOR A TOTAL OF 16 (10) BLOCKS.
2641                                ;
2642                                ;INITIALIZE LOW ADDRESS
2643                                ;SETUP AND ENABLE MMU
2644                                ;INITIALIZE BLOCK COUNTER
```

```
                                ;DO UNTIL ALL BLOCKS TESTED (16 TIMES)
                                ;.      FLUSH CACHE AND SET DIAGNOSTIC BIT
                                ;.      LET CURRENT ADDRESS = LOW ADDRESS
                                ;.      INITIALIZE ALLOCATION COUNTER
                                ;.      DO UNTIL ENTIRE BLOCK ALLOCATED (256 TIMES)
                                ;.      .       IF CURRENT ADDRESS < 32K THEN
                                ;.      .       .       READ CURRENT ADDRESS
                                ;.      .       ELSE
                                ;.      .       .       WRITE CURRENT ADDRESS
                                ;.      .       ENDIF
                                ;.      .       UPDATE CURRENT ADDRESS (ALSO ADD 200 TO PAR)
                                ;.      ENDDO
                                ;.      INITIALIZE GOOD ADDRESS
                                ;.      INITIALIZE CURRENT ADDRESS
                                ;.      INITIALIZE LOCATION COUNTER
                                ;.      SAVE CONTENTS OF VECTOR 114
                                ;.      LET VECTOR 114 POINT TO TAG STORE PARITY ABORT ROUTINE
                                ;.      DO UNTIL ALL LOCATIONS CHECKED (256 TIMES)
                                ;.      .       INITIALIZE CHECK COUNTER
                                ;.      .       DO UNTIL ADDRESS IN EACH BLOCK CHECKED
                                ;.      .       .       READ CURRENT ADDRESS
                                ;.      .       .       IF HIT THEN
                                ;.      .       .       .       IF CURRENT ADDRESS NE GOOD ADDRESS THEN
                                ;.      .       .       .       .       ERROR IN CACHE SYSTEM
                                ;.      .       .       .       ENDIF
                                ;.      .       .       ELSE
                                ;.      .       .       .       IF CURRENT ADDRESS EQUAL GOOD ADDRESS THEN
                                ;.      .       .       .       .       ERROR IN CACHE SYSTEM
                                ;.      .       .       .       ENDIF
                                ;.      .       .       ENDIF
                                ;.      .       .       UPDATE GOOD ADDRESS (ADD #2)
                                ;.      .       .       UPDATE CURRENT ADDRESS
                                ;.      .       ENDDO
                                ;.      .       UPDATE LOW ADDRESS (ADD #2000)
                                ;.      ENDDO
                                ;ENDDO
                                ;DISABLE MMU
                                ;RESTORE VECTOR 114
                                ;ENDTST

                                ;R3=GOODAD
                                ;R5=CURADD
                                ;;*****************************************************************
007274                          TST21:
007274  005267  171504                  INC     $TESTN                  ;INCREMENT TEST NUMBER
007300  032777  000400  171540          BIT     #BIT08,@SWR             ;IS THIS AN 18 BIT SYSTEM?
007306  001402                          BEQ     300$                    ;DO THIS TEST IF IT ISN'T
007310  000167  000404                  JMP     10$                     ;ELSE: JUMP TO END OF TEST
007314  012767  140000  171564  300$:   MOV     #140000,LOWADD          ;INITIALIZE LOW ADDRESS (USE PAR6)
007322  004767  172064                  JSR     PC,     INITMM          ;INITIALIZE MMU
007326  012767  000020  163162          MOV     #20,SR3                 ;ENABLE 22 BIT ADDRESSING
007334  052737  000001  177572          BIS     #BIT00,@#SR0            ;ENABLE MMU
007342  012767  177760  171510          MOV     #-16.,  LOOPIN          ;DO UNTIL ALL BLOCKS TESTED
007350  012737  002000  172354  1$:     MOV     #2000,@#KIPAR6          ;SET KIPAR6 RELOCATION CONSTANT TO PAGE 8
007356  012737  000402  177746          MOV     #402,   @#CCR           ;FLUSH CACHE AND SET DIAG BIT
007364  016705  171516                  MOV     LOWADD, R5              ;GET FIRST ADDRESS IN CURRENT BLOCK
```

```
2701  007370  012767  177400  171460          MOV     #-256., ALLCTR          ;DO UNTIL ALL ADDRESSES ALLOCATED
2702  007376  005015                  3$:     CLR     (R5)                    ;WRITE CURRENT ADDRESS
2703  007400  062705  000002          4$:     ADD     #2,     R5              ;UPDATE CURRENT ADDRESS
2704  007404  062737  000200  172354          ADD     #200,   @#KIPAR6        ;ADD 200 TO KIPAR6 TO INC TAG CONTENTS
2705  007412  005267  171440                  INC     ALLCTR                  ;IF ALL ADDRESSES ALLOCATED
2706  007416  001367                          BNE     3$                      ;ENDDO
2707
2708                                  ;;ADDRESSES HAVE NOW BEEN ALLOCATED. NEXT WE'LL CHECK THAT
2709                                  ;;THERE WAS NO DUAL ADDRESSING.
2710
2711  007420  016703  171462                  MOV     LOWADD, R3              ;INITIALIZE GOOD ADDRESS
2712  007424  012705  140000                  MOV     #140000,R5              ;GET FIRST ADDRESS
2713  007430  012737  002000  172354          MOV     #2000,@#KIPAR6          ;RESET KIPAR6
2714  007436  013767  000114  171364          MOV     @#114,  SLOC00          ;SAVE VECTOR
2715  007444  012737  001630  000114          MOV     #RAMPAR,@#114           ;LET VECTOR POINT TO RAM ABORT ROUTINE
2716
2717  007452  012767  177400  171376          MOV     #-256., ALLCTR          ;DO UNTIL ALL LOCATIONS CHECKED
2718  007460  012767  177761  171366  5$:     MOV     #-15.,  DCOUNT          ;DO UNTIL ADDRESS IN EACH BLOCK CHECKED
2719  007466  005015                  6$:     CLR     (R5)                    ;WRITE R5
2720  007470  013702  177752          205$:   MOV     @#HITMIS,R2             ;SAVE THE CONTENTS OF THE HIT/MISS REG
2721
2722                                  ;THE FOLLOWING CODE IS TO CHECK FOR EXTRA FETCHES CAUSED BY THE PRE-FETCH
2723                                  ;BUFFER BEING OVER WRITTEN WHEN BITS 1-5 OF THE PREVIOUS WRITE
2724                                  ;ADDRESS (THE EFFECTIVE ADDRESS OF CLR (R5)) ARE EQUAL TO BITS 1-5
2725                                  ;OF THE CURRENT INSTRUCTION FETCH ADDRESS. (ADDRESS OF MOV @#HITMIS,R2 INST)
2726                                  ;THE EXTRA FETCH CAUSES THE HIT MISS REGISTER CONTENTS TO BE SHIFTED.
2727                                  ;
2728  007474  012704  007470                  MOV     #205$,R4                ;SAVE ADDRESS OF MOVE INST IN R4
2729  007500  005104                          COM     R4                      ;COMPLIMENT R4
2730  007502  074504                          XOR     R5,R4                   ;IF ANY OF BITS 1-5 IN R5 ARE EQUAL
2731                                                                          ;TO ANY OF BITS 1-5 IN R4 THEN THE
2732                                                                          ;ADDRESSES WERE NOT EQUAL
2733  007504  042704  177701                  BIC     #177701,R4              ;ONLY CONCERNED WITH BITS 5-1
2734  007510  022704  000076                  CMP     #76,R4                  ;IF R4=76 THEN BITS 5-1 IN R4 AND R5
2735                                                                          ;WERE COMPLIMENTS
2736  007514  001404                          BEQ     220$                    ;IF THEY'RE NOT
2737  007516  032702  000004                  BIT     #BIT02,R2               ;THEN CHECK FOR BIT2 SET IN HIT/MISS REG.
2738  007522  001416                          BEQ     7$                      ;IF ACCESS WAS A CACHE MISS THEN GO TO 7$
2739  007524  000403                          BR      230$                    ;ELSE IT WAS A HIT; GO TO 230$
2740  007526  032702  000010          220$:   BIT     #BIT03,R2               ;ELSE CHECK FOR BIT3 SET IN HIT/MISS REG.
2741  007532  001412                          BEQ     7$                      ;IF ITS NOT A HIT THEN GO TO 7$
2742                                                                          ;ELSE
2743  007534  020503          230$:   CMP     R5,     R3                      ;IF CURRENT ADDRESS NOT EQUAL TO GOOD
2744  007536  001422                          BEQ     8$                      ;ADDRESS THEN
2745  007540  012737  007300  001120          MOV     #TST21+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
2746  007546  004767  172066                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
2747                                                                                  ;CACHE SYSTEM ERROR
2748                                                                                  ;GOT A CACHE MISS WHEN EXPECTING A HIT.
2749  007552  104000                  ERROR                                   ;ALL ERRORS TO TRAP TO EMT VECTOR
2750  007554  000072                  .WORD   72                              ;UNIQUE ERROR NUMBER
2751  007556  001325                  .WORD   ERRMSG                          ;ADDRESS OF ERROR MESSAGE
2752  007560  020503          7$:     CMP     R5,     R3                      ;IF CURRENT ADDRESS EQUAL GOOD ADDRESS
2753  007562  001010                  BNE     8$                              ;THEN
2754  007564  012737  007300  001120          MOV     #TST21+4,@#SOFTRE       ;MOVE RETRY ADDRESS TO SOFT RETURN
2755  007572  004767  172042                  JSR     PC,APTSFT               ;GO TO SOFT ERROR SUBROUTINE
2756                                                                                  ;CACHE SYSTEM ERROR
```

```
2757
2758  007576  104000                        ERROR
2759  007600  000073                        .WORD   73                      ;GOT A CACHE HIT WHEN EXPECTING A MISS.
2760  007602  001325                        .WORD   ERRMSG                  ;ALL ERRORS TO TRAP TO EMT VECTOR
2761  007604  062705  001000          8$:   ADD     #1000,  R5                ;UNIQUE ERROR NUMBER
2762  007610  005267  171240                INC     DCOUNT                  ;ADDRESS OF ERROR MESSAGE
2763  007614  002724                        BLT     6$                      ;UPDATE TO NEXT BLOCK
2764  007616  062703  000002                ADD     #2,     R3              ;IF ADDRESS NOT CHECKED FOR EACH BLOCK
2765  007622  042705  017000                BIC     #17000, R5              ;ENDDO
2766  007626  062705  000002                ADD     #2,     R5              ;UPDATE GOOD ADDRESS
2767  007632  062737  000200  172354        ADD     #200,   @#KIPAR6        ;UPDATE ADDRESS TO BE CHECKED
2768  007640  005267  171212                INC     ALLCTR
2769  007644  002705                        BLT     5$                      ;IF ALL ADDRESSES CHECKED
2770  007646  062767  001000  171232        ADD     #1000,  LOWADD          ;ENDDO
2771  007654  005267  171200                INC     LOOPIN                  ;UPDATE TO NEXT BLOCK TO BE ALLOCATED
2772  007660  002002                        BGE     9$                      ;IF ALL BLOCKS WERE TESTED
2773  007662  000167  177462                JMP     1$                      ;ENDDO
2774  007666  005067  167700          9$:   CLR     SR0                     ;
2775  007672  016737  171132  000114        MOV     SLOC00, @#114           ;DISABLE MMU
2776  007700  005037  177744                CLR     @#MSER                  ;RESTORE ABORT VECTOR
2777  007704  005037  177766                CLR     @#CPEREG                ;CLEAR THE MEM SYS ERROR REG
2778  007710  005037  001116                CLR     @#SOFTER                ;CLEAR THE CPU ERROR REGISTER
2779  007714  000167  000000                JMP     TST22                   ;CLEAR THE SOFT ERROR COUNT
2780                                                                        ;JUMP OVER DATA TABLES OR SUBROUTINES
2781  007720                          10$:                                  ;TO NEXT TEST
2782
```

```
                                        .DSABL  AMA

007720                                  CTST24:
                                        ;;*****************************************************************
                                        ;*TEST 22       TAG RAM DATA RELIABILITY TEST
                                        ;;*****************************************************************
007720                                  TST22:
007720  005267  171060                          INC     $TESTN                  ;INCREMENT TEST NUMBER
007724  032777  001000  171114                  BIT     #BIT09,@SWR             ;ARE DATA RELIABILITY TESTS SELECTED?
007732  001002                                  BNE     1000$                   ;IF BIT09 IN SWR = 1 ;THEN DO TEST
007734  000167  000772                          JMP     XXX                     ;ELSE GO TO END.
007740  032777  000400  171100  1000$:  BIT     #BIT08,@SWR             ;IS THIS AN 18 BIT SYSTEM?
007746  001404                                  BEQ     1001$                   ;IF BIT08 IN SWR=0 BRANCH TO 1001$
007750  012767  177777  171064                  MOV     #177777,FLAG            ;SET UP INDICATOR FOR 18 BIT SYS
007756  000402                                  BR      1002$                   ;
007760  005067  171056          1001$:  CLR     FLAG                    ;SET UP INDICATOR FOR 22 BIT SYSTEM
007764  013767  000114  171036  1002$:  MOV     @#114,SLOC00            ;SAVE OLD PARITY ERROR VECTOR
007772  012737  000116  000114                  MOV     #116,@#114              ;PUT TRAP CATCHER IN FOR CACHE PARITY ERRORS
010000  005037  000116                          CLR     @#116                   ;
010004  004767  171402                          JSR     PC,     INITMM          ;INITIALIZE MMU
010010  042767  100000  162274                  BIC     #BIT15,KIPDR5           ;CLEAR CACHE BYPASS ON PDR5
010016  012767  000020  162472                  MOV     #20,SR3                 ;ENABLE 22 BIT ADDRESSING.
010024  012767  001600  162316                  MOV     #1600,KIPAR4            ;SET KIPAR4 TO RELOCATE TO BANK 7 OF MEM.
010032  012701  100000                          MOV     #100000,R1              ;SET UP R1 AS POINTER TO BANK 7 VIA KIPAR4
010036  012702  000000                          MOV     #0,R2                   ;SET UP R2 AS POINTER TO BANK 0
010042  012700  010000                          MOV     #4096.,R0               ;SET UP R0 AS COUNTER
010046  012767  000001  167516                  MOV     #1,SR0                  ;TURN ON MMU
010054  012221                  100$:   MOV     (R2)+,(R1)+             ;MOVE ALL THE CODE IN FIRST 4KW TO BANK 7
010056  077002                          SOB     R0,100$                 ;
010060  000137  110064                  JMP     @#101$+100000           ;START EXECUTING OUT OF BANK 7
                                        ;
                                        ;WE ARE NOW OPERATING FROM PHYSICAL MEMORY IN BANK 7 (160000-177776)
                                        ;
010064  012701  140000          101$:   MOV     #140000,R1              ;SETUP TO INIT CACHE TAG STORE TO ZEROS
010070  005037  172354                  CLR     @#KIPAR6                ;SET KIPAR6 TO RELOCATE TO PAGE 0
010074  042737  100000  172300          BIC     #BIT15,@#KIPDR0         ;ALLOW CACHE REFERENCE ON PAGE ZERO
010102  012702  010000                  MOV     #10000, R2              ;INITIALIZE 4K WORDS
010106  005721                  1$:     TST     (R1)+                   ;CLEAR TAG STORE
010110  077202                          SOB     R2,     1$              ;LOOP UNTIL ALL OF TAG STORE INITIALIZED
                                        ;
010112  012737  000002  177746          MOV     #BIT01,@#CCR            ;SET DIAGNOSTIC BIT IN CCR
010120  012701  110672                  MOV     #TAGAD1+100000,R1       ;MAKE R1 A POINTER TO TABLE OF DATA.
010124  012706  120000                  MOV     #120000,R6              ;SET UP TEST ADDRESS IN R6
                                                                        ;USING R6 TO SPEED UP TEST LOOP
010130  012702  140002                  MOV     #140002,R2              ;R2 WILL BE USED TO INDEX THRU THE CACHE.
                                        ;START TESTING
                                        ;THE OBJECT OF THIS TEST IS TO TRY EVERY POSIBLE DATA COMBINATION AT A
                                        ;GIVEN ADDRESS WHILE ALL OTHER ADDRESSES ARE SET TO ALL ZEROS. ONCE A
                                        ;CHANGE IN THE DATA AT A PARTICULAR ADDRESS HAS BEEN MADE; ALL OTHER
                                        ;ADDRESSES ARE CHECKED TO ENSURE THAT DATA HAS NOT BEEN ALTERED. WHEN ALL
                                        ;DATA COMBINATIONS HAVE BEEN CHECKED AT A PARTICULAR ADDRESS; THAT ADDRESS
                                        ;IS RESET TO ZEROS AND THE NEXT CONSECUTIVE ADRESS IS TESTED. THIS IS DONE
                                        ;UNTIL ALL 4096 LOCATIONS HAVE BEEN TESTED.

010134  012137  172352          FLD0:   MOV     (R1)+,@#KIPAR5          ;MOVE RELOCATION OFFSET (DATA THAT WILL
```

```
2839
2840  010140  005767  170676          TST     FLAG            ;BE IN THE TAG RAMS) TO KIPAR5.
2841  010144  001403                  BEQ     1$              ;IS THIS AN 18 BIT SYSTEM
2842  010146  042737  170000  172352  BIC     #170000,@#KIPAR5 ;BRANCH IF NOT
2843  010154  005016          1$:     CLR     (R6)            ;MODIFY FOR 18 BIT SYSTEM
2844  010156  013703  177752          MOV     @#HITMIS,R3     ;LOAD TAG RAMS WITH DATA PATTERN.
2845  010162  005703                  TST     R3              ;SAVE CONTENTS OF HIT/MISS REG IN R3
2846  010164  001403                  BEQ     FLD0A           ;HIT MISS REGISTER SHOULD BE CLEAR
2847  010166  104000                  ERROR                   ;BRANCH TO FLD0A
2848  010170  000074                  .WORD   74              ;ALL ERRORS TO TRAP TO EMT VECTOR
2849  010172  001325                  .WORD   ERRMSG          ;UNIQUE ERROR NUMBER
2850                                                          ;ADDRESS OF ERROR MESSAGE
2851                                                                  ;CACHE SYSTEM ERROR
2852  010174                  FLD0A:                          ;ERROR!! WE GOT A CACHE HIT- WANTED A MISS
2853  010174  005716                  TST     (R6)
2854  010176  032737  000010  177752  BIT     #BIT03,@#HITMIS ;CHECK THAT CACHE WAS PROPERLY ALLOCATED.
2855  010204  001003                  BNE     100$            ;WE'RE LOOKING FOR A HIT THIS TIME.
2856  010206  104000                  ERROR                   ;
2857  010210  000075                  .WORD   75              ;ALL ERRORS TO TRAP TO EMT VECTOR
2858  010212  001325                  .WORD   ERRMSG          ;UNIQUE ERROR NUMBER
2859                                                          ;ADDRESS OF ERROR MESSAGE
2860                                                                  ;CACHE SYSTEM ERROR
2861  010214  012700  010000  100$:   MOV     #4096.,R0       ;ERROR!
2862  010220  012704  000004          MOV     #BIT02,R4       ;INIT R0 TO COUNT 4KWORDS.
2863  010224  012705  160000          MOV     #160000,R5      ;LOAD DATA INTO R4 TO SPEED UP BIT INST.
2864  010230  020502          1$:     CMP     R5,R2           ;LOAD DATA INTO R5 TO SPEED UP CMP INST.
2865  010232  001410                  BEQ     102$            ;IF TOP OF CACHE HAS BEEN REACHED
2866  010234  005722          2$:     TST     (R2)+           ;GO RESET R2 TO 140000
2867                                                          ;ENSURE THAT OTHER CACHE LOCATIONS
2868  010236  030437  177752          BIT     R4,@#HITMIS     ;WERE UNAFFECTED.
2869  010242  001407                  BEQ     3$              ;ALL CACHE LOCATIONS SHOULD HIT.
2870  010244  077007          101$:   SOB     R0,1$           ;KEEP GOING UNTIL WE GET A MISS
2871  010246  104000                  ERROR                   ;TEST THE NEXT CACHE LOCATION
2872  010250  000076                  .WORD   76              ;ALL ERRORS TO TRAP TO EMT VECTOR
2873  010252  001325                  .WORD   ERRMSG          ;UNIQUE ERROR NUMBER
2874                                                          ;ADDRESS OF ERROR MESSAGE
2875                                                                  ;CACHE SYSTEM ERROR
2876                                                          ;ERROR! WE SHOULD HAVE GOTTEN A MISS
2877  010254  042702  037776  102$:   BIC     #37776,R2       ;BEFORE FALLING THRU SOB
2878  010260  000765                  BR      2$              ;RESET R2 TO 140000
2879  010262  010203          3$:     MOV     R2,R3           ;
2880  010264  162703  020002          SUB     #20002,R3       ;SAVE CONTENTS OF R2 IN R3
2881  010270  020306                  CMP     R3,R6           ;SUBTRACT 20002 FROM CONTENTS OF R3
2882                                                          ;SHOULD BE EQUAL. IF NOT; THEN THE
2883  010272  001403                  BEQ     4$              ;CACHE RAM HAS BEEN ALTERED
2884  010274  104000                  ERROR                   ;GO TO NEXT TEST
2885  010276  000077                  .WORD   77              ;ALL ERRORS TO TRAP TO EMT VECTOR
2886  010300  001325                  .WORD   ERRMSG          ;UNIQUE ERROR NUMBER
2887                                                          ;ADDRESS OF ERROR MESSAGE
2888  010302  005711          4$:     TST     (R1)                    ;CACHE SYSTEM ERROR
2889  010304  001313                  BNE     FLD0            ;HAVE WE TRIED ALL PATTERNS AT THIS LOC?
2890                                                          ;GO GET NEXT ADDRESS IF YES.
2891  010306  022706  137776  5$:     CMP     #137776,R6      ;IF NOT GO DO THE NEXT DATA PATTERN.
2892  010312  001410                  BEQ     FLD1            ;HAVE WE TESTED ALL 4KW CACHE LOCATIONS?
2893  010314  062706  000002          ADD     #2,R6           ;IF YES THEN GO TO NEXT TEST
2894  010320  062702  000002          ADD     #2,R2           ;ELSE GET THE NEXT ADDRESS...
                                                              ;BUMP R2
```

```
2895  010324  012701  110672          MOV     #TAGAD1+100000,R1       ;RESET DATA POINTER
2896  010330  000167  177600          JMP     FLD0                    ;DO IT AGAIN
2897                                  ;
2898  010334  012701  140000  FLD1:   MOV     #140000,R1              ;INIT R1
2899  010340  012737  077600  172354  MOV     #77600,@#KIPAR6         ;DATA IN KIPAR6 WILL CAUSE TAG RAMS
2900                                                                  ;TO BE SET TO ALL 1'S.
2901  010346  005767  170470          TST     FLAG                    ;IS THIS AN 18 BIT SYSTEM?
2902  010352  001406                  BEQ     100$                    ;BRANCH IF NOT
2903  010354  042737  170000  172354  BIC     #170000,@#KIPAR6        ;MODIFY FOR 18 BIT SYSTEM
2904  010362  012737  000002  177746  MOV     #BIT01,@#CCR            ;SET DIAGNOSTIC BIT IN CACHE CONTROL REG
2905  010370  012702  010000  100$:   MOV     #4096.,R2               ;SET UP TO FILL 4KW OF CACHE WITH ONES
2906  010374  005021          1$:     CLR     (R1)+                   ;WRITE THE CACHE
2907  010376  077202                  SOB     R2,1$                   ;
2908  010400  012701  110704          MOV     #TAGAD2+100000,R1       ;SET UP R1 AS A POINTER TO DATA TABLE
2909  010404  012706  120000          MOV     #120000,R6              ;SET UP TESTADDRESS
2910  010410  012702  140002          MOV     #140002,R2              ;SET UP R2
2911  010414  012137  172352  FLD1A:  MOV     (R1)+,@#KIPAR5          ;DATA IN KIPAR5 WILL BE WRITTEN INTO TAG RAM
2912  010420  005767  170416          TST     FLAG                    ;IS THIS AN 18 BIT SYSTEM?
2913  010424  001403                  BEQ     1$                      ;BRANCHIF NOT
2914  010426  042737  170000  172352  BIC     #170000,@#KIPAR5        ;MODIFY FOR 18 BIT SYSTEM
2915  010434  005016          1$:     CLR     (R6)                    ;WRITE TAG RAM WITH TEST DATA
2916  010436  013703  177752          MOV     @#HITMIS,R3             ;STORE CONTENTS OF HIS MISS REGISTER
2917  010442  005703                  TST     R3                      ;HIT/MISS REGISTER SHOULD BE CLEAR
2918  010444  001403                  BEQ     FLD1B                   ;BRANCH IF IT WAS A MISS
2919  010446  104000                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2920  010450  000100                  .WORD   100                     ;UNIQUE ERROR NUMBER
2921  010452  001325                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2922                                                                                  ;CACHE SYSTEM ERROR
2923  010454                  FLD1B:
2924  010454  005716          1$:     TST     (R6)                    ;TEST ADDRESS SHOULD NOW BE ALLOCATED IN CACHE
2925  010456  032737  000010  177752  BIT     #BIT03,@#HITMIS         ;DID IT HIT?
2926  010464  001003                  BNE     100$                    ;BRANCH IF YES
2927  010466  104000                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2928  010470  000101                  .WORD   101                     ;UNIQUE ERROR NUMBER
2929  010472  001325                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2930                                                                                  ;CACHE SYSTEM ERROR
2931  010474  012700  010000  100$:   MOV     #4096.,R0               ;INIT R0 TO COUNT 4 KWORDS
2932  010500  012705  160000          MOV     #160000,R5              ;STORE DATA IN R5 FOR FASTER CMP INST.
2933  010504  020502          2$:     CMP     R5,R2                   ;IF THE TOP OF CACHE HAS BEEN REACHED
2934  010506  001412                  BEQ     104$                    ;THEN GO RESET R2 TO 140000
2935  010510  005022          101$:   CLR     (R2)+                   ;CHECK THE OTHER CACHE LOCATIONS TO
2936                                                                  ;SEE THAT THEY STILL HIT.
2937  010512  013703  177752  102$:   MOV     @#HITMIS,R3             ;SAVE HIT MISS REG IN R3
2938  010516  032703  000004          BIT     #BIT02,R3               ;DID IT HIT?
2939  010522  001407                  BEQ     105$                    ;KEEP TESTING UNTIL A MISS OCCURS
2940  010524  077011          103$:   SOB     R0,2$                   ;TEST THE NEXT CACHE LOCATION
2941  010526  104000                  ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2942  010530  000102                  .WORD   102                     ;UNIQUE ERROR NUMBER
2943  010532  001325                  .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2944                                                                                  ;CACHE SYSTEM ERROR
2945                                                                  ;ERROR! SHOULD NOT HAVE FALLEN THRU
2946                                                                  ;SOB INST BEFORE GETTING A MISS
2947  010534  042702  037776  104$:   BIC     #37776,R2               ;RESET R2 TO 140000
2948  010540  000763                  BR      101$                    ;
2949  010542  012704  010512  105$:   MOV     #102$,R4                ;SAVE ADDR OF MOV INSTRUCTION IN R4
2950  010546  010267  170332          MOV     R2,CURADD               ;SAVE R2 IN CURADD
```

```
2951  010552  162767  000002  170324          SUB     #2,CURADD               ;SUBTRACT 2 FROM R2
2952  010560  042767  177701  170316          BIC     #177701,CURADD          ;ONLY INTERESTED IN BITS 1-5
2953  010566  042704  177701                  BIC     #177701,R4              ;STRIP OFF UN-NEEDED BITS
2954  010572  020467  170306                  CMP     R4,CURADD               ;IF BITS 5-1IN R4 AND CURADD ARE EQUAL?
2955  010576  001004                          BNE     3$                      ;THEN
2956  010600  032703  000010                  BIT     #BIT03,R3               ;TEST BIT03 IN HIT MISS REG
2957  010604  001401                          BEQ     3$                      ;IF ITS A HIT THEN DO THE NEXT ADDR
2958  010606  000746                          BR      103$                    ;
2959  010610  010203                  3$:     MOV     R2,R3                   ;STORE CONTENTS OF R2 IN R3
2960  010612  162703  020002                  SUB     #20002,R3               ;SUBTRACT 20002 FROM R3
2961  010616  020306                          CMP     R3,R6                   ;IS R3= TO THE TEST ADDRESS.
2962  010620  001403                          BEQ     4$                      ;BRANCH IF THEY ARE EQUAL
2963  010622  104000                          ERROR                           ;ALL ERRORS TO TRAP TO EMT VECTOR
2964  010624  000103                          .WORD   103                     ;UNIQUE ERROR NUMBER
2965  010626  001325                          .WORD   ERRMSG                  ;ADDRESS OF ERROR MESSAGE
2966                                                                                  ;CACHE SYSTEM ERROR
2967  010630  005711                  4$:     TST     (R1)                    ;WAS THIS THE LAST PATTERN FOR THIS LOCATION?
2968  010632  001400                          BEQ     6$                      ;GO GET NEXT ADDRESS IF YES
2969                                  ;
2970                                  ;CHECK TO SEE IF ALL LOCATIONS HAVE BEEN TESTED. IF NOT, GENERATE THE NEW
2971                                  ;ADDRESS, RESET THE DATA POINTER AND TEST THE NEXT CACHE LOCATION.
2972                                  ;
2973  010634  022706  137776          6$:     CMP     #137776,R6              ;HAVE WE TESTED ALL 4KW CACHE LOCATIONS?
2974  010640  001412                          BEQ     8$                      ;IF YES THEN GO TO END OF TEST
2975  010642  005062  177776          7$:     CLR     -2(R2)                  ;ELSE WRITE ONES IN THE LOCATION LAST TESTED
2976  010646  062706  000002                  ADD     #2,R6                   ;GET THE NEXT ADDRESS...
2977  010652  062702  000002                  ADD     #2,R2                   ;BUMP R2
2978  010656  012701  110704                  MOV     @TAGAD2+100000,R1       ;RESET DATA POINTER
2979  010662  000167  177526                  JMP     FLD1A                   ;DO IT AGAIN
2980  010666  000137  010716          8$:     JMP     @#TSEND                 ;JUMP OUT OF BANK 7
2981                                  ;
2982                                  ;
2983  010672  104200                  TAGAD1: .WORD   104200
2984  010674  010400                          .WORD   010400
2985  010676  121000                          .WORD   121000
2986  010700  042000                          .WORD   042000
2987  010702  000000                          .WORD   000000
2988  010704  073400                  TAGAD2: .WORD   073400
2989  010706  167200                          .WORD   167200
2990  010710  056600                          .WORD   056600
2991  010712  135600                          .WORD   135600
2992  010714  000000                          .WORD   000000
2993                                  ;
2994  010716  005037  177746          TSEND:  CLR     @#CCR                   ;CLEAR THE CCR
2995  010722  005037  177572                  CLR     @#SR0                   ;TURN OFF MMU
2996  010726  012706  001000                  MOV     #STBOT,R6               ;RESTORE STACK POINTER
2997  010732  016737  170072  000114  XXX:    MOV     SLOC00, @#114           ;RESTORE VECTOR
2998  010740  016737  170066  000116          MOV     SLOC01, @#116
2999
3000
3001
```

```
                                        .MCALL  IDMSG,ENDPAS
                                        .SBTTL  END OF PASS ROUTINE

                                        ;;**********************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO RESTART

011746                                  $EOP:
011746  005767  170034                          TST     $PASS                   ;ONLY TYPE MESSAGE AT END OF FIRST PASS
011752  001002                                  BNE     SKIPID                  ;IF >0 THEN SKIP THE ID MESSAGE
011754  104401  011054                          TYPE    ,MSG1                   ;ELSE TYPE THE ID MESSAGE
011760                                  SKIPID:
011760  005267  170022                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
011764  042767  100000  170014                BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011772  005327                                  DEC     (PC)+           ;;LOOP?
011774  000001                          $EOPCT: .WORD   1
011776  003022                                  BGT     $DOAGN          ;;YES
012000  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
012002  000001                          $ENDCT: .WORD   1
012004  010774                                  $EOPCT
012006  104401  011135                          TYPE    ,MSG2
012012  016746  167770                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
012016  104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
012020  104401  011050                          TYPE    ,$ENULL
012024  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
012030  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
012032  000005                                  RESET                   ;;CLEAR THE WORLD
012034  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
012036  000240                                  NOP                     ;;SAVE ROOM
012040  000240                                  NOP                     ;;FOR
012042  000240                                  NOP                     ;;ACT11
012044                                  $DOAGN:
012044  000137                                  JMP     @(PC)+          ;;RETURN
012046  002176                          $RTNAD: .WORD   RESTART
012050     377     377     000         $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
                012054                          .EVEN
012054  005015  055103  042113          MSG1:   .ASCIZ  <CR><LF>/CZKDM-B-0 KDJ11 CACHE MEMORY SYSTEM DIAGNOSTIC/
012062  026515  026502  020060
012070  042113  030512  020061
012076  040503  044103  020105
012104  042515  047515  054522
012112  051440  051531  042524
012120  020115  044504  043501
012126  047516  052123  041511
012134     000
012135     015  041412  045532          MSG2:   .ASCIZ  <CR><LF>/CZKDMB END PASS #/
012142  046504  020102  047105
012150  020104  040520  051523
012156  021440     000
                012162                  .EVEN
                                        .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
                                        ;*
011162  105767  000343          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
011166  100002                          BPL     1$              ;;BR IF YES
011170  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
011172  000430                          BR      3$              ;;LEAVE
011174  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
011176  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
011202  122767  000001  167610          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
011210  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
011212  132767  000100  167601          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
011220  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
011222  010067  000004                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
011226  004767  001622                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
011232  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
011234  132767  000040  167557  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
011242  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
011244  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
011246  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
011250  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
011252  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
011254  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
011260  000002                          RTI                     ;;RETURN
011262  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
011266  001430                          BEQ     8$
011270  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
011274  001006                          BNE     5$
011276  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
011300  104401                          TYPE                    ;;TYPE A CR AND LF
011302  001403                          $CRLF
011304  105067  000202                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
011310  000755                          BR      2$              ;;GET NEXT CHARACTER
011312  004767  000056          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
011316  126726  000206          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
011322  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
011324  016746  000176                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
011330  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
011334  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
011336  004767  000032                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
011342  105367  000144                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
011346  000770                          BR      7$              ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR
```

```
3114  011350  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3115  011354  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3116  011360  132767  000007  000124          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3117  011366  001372                          BNE     9$              ;;TAB STOP
3118  011370  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
3119  011372  000724                          BR      2$              ;;GET NEXT CHARACTER
3120  011374                        $TYPEC:
3121  011374  105777  000116                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?          ;MJD001
3122  011400  100022                          BPL     10$             ;;BR IF NOT                     ;MJD001
3123  011402  017746  000112                  MOV     @$TKB,-(SP)     ;;GET CHAR                      ;MJD001
3124  011406  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS         ;MJD001
3125  011412  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF                 ;MJD001
3126  011416  001012                          BNE     102$            ;;BR IF NOT                     ;MJD001
3127  011420                        101$:                                                             ;MJD001
3128  011420  105777  000072                  TSTB    @$TKS           ;;WAIT FOR CHAR                 ;MJD001
3129  011424  100375                          BPL     101$                                            ;MJD001
3130  011426  117716  000066                  MOVB    @$TKB,(SP)      ;;GET CHAR                      ;MJD001
3131  011432  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT                      ;MJD001
3132  011436  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?                   ;MJD001
3133  011442  001366                          BNE     101$            ;;BR IF NOT                     ;MJD001
3134  011444                        102$:                                                             ;MJD001
3135  011444  005726                          TST     (SP)+           ;;FIX STACK                     ;MJD001
3136  011446                        10$:                                                              ;MJD001
3137  011446  105777  000050                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3138  011452  100375                          BPL     10$                                             ;MJD001
3139  011454  116677  000002  000042          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3140  011462  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3141  011470  001003                          BNE     1$              ;;BRANCH IF NO
3142  011472  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3143  011476  000406                          BR      $TYPEX          ;;EXIT
3144  011500  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3145  011506  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
3146  011510  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
3147  011512  000000              $CHARCNT:.WORD  0                   ;;CHARACTER COUNT STORAGE
3148  011514  000207              $TYPEX: RTS     PC
3149
3150  011516  177560              $TKS:   .WORD   177560          ;;TTY KDB STATUS                ;MJD001
3151  011520  177562              $TKB:   .WORD   177562          ;;TTY KBD BUFFER                ;MJD001
3152  011522  177564              $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
3153  011524  177566              $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
3154  011526     000              $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
3155  011527     002              $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
3156  011530     012              $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
3157  011531     000              $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
3158  011532     077              $QUES:  .ASCII  "?"             ;;QUESTION MARK
3159  011533     012      000     $LF:    .ASCIZ  <12>            ;;LINEFEED
3160          011536              .EVEN
3161                      .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3162
3163                      ;;**************************************************************************
3164                      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3165                      ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
3166                      ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3167                      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3168                      ;*REPLACED WITH SPACES.
3169                      ;*CALL:
```

```
                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                ;*      TYPDS                   ;;GO TO THE ROUTINE
                                $TYPDS:
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                                        MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
                                        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
                                        BPL     1$              ;;BR IF INPUT IS POS.
                                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
                                        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
                                1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
                                        MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
                                        MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
                                2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
                                        MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
                                3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
                                        BLT     4$              ;;BR IF DONE
                                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
                                        BR      3$
                                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
                                        TST     R2              ;;CHECK IF BCD DIGIT=0
                                        BNE     5$              ;;FALL THROUGH IF 0
                                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
                                        BMI     7$              ;;BR IF YES
                                5$:     ASLB    (SP)            ;;MSD?
                                        BCC     6$              ;;BR IF NO
                                        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
                                6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
                                7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
                                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
                                        TST     (R0)+           ;;JUST INCREMENTING
                                        CMP     R0,#10          ;;CHECK THE TABLE INDEX
                                        BLT     2$              ;;GO DO THE NEXT DIGIT
                                        BGT     8$              ;;GO TO EXIT
                                        MOV     R5,R2           ;;GET THE LSD
                                        BR      6$              ;;GO CHANGE TO ASCII
                                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
                                        BPL     9$              ;;BR IF NO
                                        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
                                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
                                        MOV     (SP)+,R5        ;;POP STACK INTO R5
                                        MOV     (SP)+,R3        ;;POP STACK INTO R3
                                        MOV     (SP)+,R2        ;;POP STACK INTO R2
                                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                                        MOV     (SP)+,R0        ;;POP STACK INTO R0
                                        TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
                                        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
                                        MOV     (SP)+,(SP)
                                        RTI                     ;;RETURN TO USER
                                $DTBL:  10000.
                                        1000.
                                        100.
```

```
                                        10.
                                $DBLK:  .BLKW   4
                                .SBTTL  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

011762  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
011766  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
011774  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
012000  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
012004  000406                          BR      $TYPON
012006  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
012014  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
012022  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
012030  010346                          MOV     R3,-(SP)        ;;SAVE R3
012032  010446                          MOV     R4,-(SP)        ;;SAVE R4
012034  010546                          MOV     R5,-(SP)        ;;SAVE R5
012036  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
012042  005404                          NEG     R4
012044  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
012050  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
012054  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
012060  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
012064  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
012066  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
012070  000404                          BR      3$              ;;GO DO MSB
012072  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
012074  006105                          ROL     R5
012076  006105                          ROL     R5
012100  010503                          MOV     R5,R3
012102  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
012104  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
012110  100016                          BPL     7$              ;;BR IF NO
012112  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
012116  001002                          BNE     4$              ;;TEST FOR 0
```

```
3282  012120  005704                          TST     R4              ;;SUPPRESS THIS 0?
3283  012122  001403                          BEQ     5$              ;;BR IF YES
3284  012124  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3285  012126  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3286  012132  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3287  012136  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
3288  012142  104401  012202                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
3289  012146  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
3290  012152  003347                          BGT     2$              ;;BR IF MORE TO DO
3291  012154  002402                          BLT     6$              ;;BR IF DONE
3292  012156  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3293  012160  000744                          BR      2$              ;;GO DO THE LAST DIGIT
3294  012162  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
3295  012164  012604                          MOV     (SP)+,R4        ;;RESTORE R4
3296  012166  012603                          MOV     (SP)+,R3        ;;RESTORE R3
3297  012170  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3298  012176  012616                          MOV     (SP)+,(SP)
3299  012200  000002                          RTI                     ;;RETURN
3300  012202     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3301  012203     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3302  012204     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3303  012205     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
3304  012206  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3305                                  .SBTTL  TTY INPUT ROUTINE
3306
3307                                  ;;******************************************************************
3308                                  .ENABL  LSB
3309
3310                                  ;;******************************************************************
3311                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
3312                                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
3313                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
3314                                  ;*WHEN OPERATING IN TTY FLAG MODE.
3315  012210  022767  000176  166630  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
3316  012216  001074                          BNE     15$             ;;BRANCH IF NO
3317  012220  105777  177272                  TSTB    @$TKS           ;;CHAR THERE?
3318  012224  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
3319  012226  117746  177266                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
3320  012232  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
3321  012236  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
3322  012242  001062                          BNE     15$             ;;NO, RETURN TO USER
3323  012244  126727  000514  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
3324  012252  001456                          BEQ     15$             ;;BRANCH IF YES
3325
3326  012254  104401  012735                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
3327  012260  104401  012742          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
3328  012264  016746  165706                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
3329  012270  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3330  012272  104401  012753                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
3331  012276  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
3332  012300  005046                          CLR     -(SP)           ;;THE NEW SWR
3333  012302  105777  177210          7$:     TSTB    @$TKS           ;;CHAR THERE?
3334  012306  100375                          BPL     7$              ;;IF NOT TRY AGAIN
3335
3336  012310  117746  177204                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
3337  012314  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
```

```
3338
3339
3340
3341  012320  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
3342  012324  001005                          BNE     10$             ;;BRANCH IF NOT
3343  012326  104401  012730                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
3344  012332  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
3345  012336  000757                          BR      19$             ;;LET'S TRY IT AGAIN
3346
3347
3348  012340  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
3349  012344  001022                          BNE     16$             ;;BRANCH IF NO
3350  012346  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
3351  012352  001403                          BEQ     11$             ;;BRANCH IF YES
3352  012354  016677  000002  166464          MOV     2(SP),@SWR      ;;SAVE NEW SWR
3353  012362  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
3354  012366  104401  001403          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
3355  012372  126727  000367  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
3356  012400  001003                          BNE     15$             ;;BRANCH IF NOT
3357  012402  012777  000100  177106          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
3358  012410  000002          15$:    RTI                             ;;RETURN
3359  012412  004767  176756          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
3360  012416  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
3361  012422  002420                          BLT     18$             ;;BRANCH IF YES
3362  012424  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
3363  012430  003015                          BGT     18$             ;;BRANCH IF YES
3364  012432  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
3365  012436  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
3366  012442  001403                          BEQ     17$             ;;BRANCH IF YES
3367  012444  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
3368  012446  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
3369  012450  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
3370  012452  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
3371  012456  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
3372  012462  000707                          BR      7$              ;;GET THE NEXT ONE
3373  012464  104401  011532          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
3374  012470  000720                          BR      20$             ;;SIMULATE CONTROL-U
3375                                  .DSABL  LSB
3376
3377
3378                                  ;;*************************************************************************
3379                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3380                                  ;*CALL:
3381                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
3382                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
3383                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
3384                                  ;
3385
3386  012472  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
3387  012474  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
3388  012502  105777  177010          1$:     TSTB    @$TKS           ;;WAIT FOR
3389  012506  100375                          BPL     1$              ;;A CHARACTER
3390  012510  117766  177004  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
3391  012516  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
3392  012524  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
3393  012532  001013                          BNE     3$              ;;BRANCH IF NO
```

```
3394  012534  105777  176756          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
3395  012540  100375                  BPL     2$              ;;LOOP UNTIL ITS THERE
3396  012542  117746  176752          MOVB    @$TKB,-(SP)     ;;GET CHARACTER
3397  012546  042716  177600          BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3398  012552  022627  000021          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
3399  012556  001366                  BNE     2$              ;;IF NOT DISCARD IT
3400  012560  000750                  BR      1$              ;;YES, RESUME
3401  012562  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
3402  012570  002407                  BLT     4$              ;;BRANCH IF YES
3403  012572  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
3404  012600  003003                  BGT     4$              ;;BRANCH IF YES
3405  012602  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
3406  012610  000002                  4$:     RTI             ;;GO BACK TO USER
3407                                  ;;**************************************************************************
3408                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3409                                  ;*CALL:
3410                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
3411                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
3412                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
3413
3414  012612  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
3415  012614  012703  012720          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
3416  012620  022703  012730          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
3417  012624  101405                  BLOS    4$              ;;BR IF YES
3418  012626  104410                  RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
3419  012630  112613                  MOVB    (SP)+,(R3)      ;;GET CHARACTER
3420  012632  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
3421  012636  001003                  BNE     3$              ;;SKIP IF NOT
3422  012640  104401  011532          4$:     TYPE    ,$QUES          ;;TYPE A '?'
3423  012644  000763                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
3424  012646  111367  000044          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
3425  012652  104401  012716                  TYPE    ,9$
3426  012656  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
3427  012662  001356                  BNE     2$              ;;LOOP IF NOT RETURN
3428  012664  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
3429  012670  104401  011533                  TYPE    ,$LF            ;;TYPE A LINE FEED
3430  012674  012603                  MOV     (SP)+,R3        ;;RESTORE R3
3431  012676  011646                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
3432  012700  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
3433  012706  012766  012720  000004          MOV     #$TTYIN,4(SP)
3434  012714  000002                  RTI                     ;;RETURN
3435  012716     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
3436  012717     000                  .BYTE   0               ;;TERMINATOR
3437  012720  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
3438  012730  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
3439  012735     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
3440  012742  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
3441  012750  020075     000
3442  012753     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
3443  012760  036440  000040
3444  012764     000                  $AUTOB: .BYTE   0       ;;AUTO MODE FLAG
3445  012765     000                  $INTAG: .BYTE   0       ;;INTERRUPT MODE FLAG
3446                                  .SBTTL  TRAP DECODER
3447
3448                                  ;;**************************************************************************
3449                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
```

```
3450                                    ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3451                                    ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3452                                    ;*GO TO THAT ROUTINE.
3453
3454  012766  010046                    $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
3455  012770  016600  000002                   MOV     2(SP),R0        ;;GET TRAP ADDRESS
3456  012774  005740                           TST     -(R0)           ;;BACKUP BY 2
3457  012776  111000                           MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
3458  013000  006300                           ASL     R0              ;;POSITION FOR INDEXING
3459  013002  016000  013022                   MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
3460  013006  000200                           RTS     R0              ;;GO TO ROUTINE
3461
3462
3463                                    ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
3464
3465  013010  011646                    $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
3466  013012  016666  000004  000002           MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
3467  013020  000002                           RTI                     ;;RESTORE THE PSW
3468
3469                                    .MACRO  SETTRAP A,B,MSG
3470                                            $$SET   A,B,\<TRAP+$TRP>,\$TRP,<MSG>
3471                                    .NLIST
3472                                    $TRP=$TRP+1
3473                                    .LIST
3474                                    .ENDM   SETTRAP
3475                                    .MACRO  $$SET   A,B,C,D,COMNT
3476                                    .IF EQ $TRP-1
3477                                    .SBTTL  TRAP TABLE
3478
3479                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3480                                    ;*BY THE "TRAP" INSTRUCTION.
3481
3482                                    ;       ROUTINE
3483                                    ;       -------
3484                                    $TRPAD: .WORD   $TRAP2
3485                                    .ENDC
3486                                    .IIF NDF GNS,.NLIST
3487                                            A=      C
3488                                    .IIF NDF GNS,.LIST
3489                                            B       ;;CALL=A        TRAP+D(C)       COMNT
3490                                    .ENDM   $$SET
3491                                    .MACRO  TRMTRP
3492                                    $TERM=.-$TRPAD
3493                                    .ENDM   TRMTRP
3494                                    .SBTTL  TRAP TABLE
3495
3496                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3497                                    ;*BY THE "TRAP" INSTRUCTION.
3498
3499                                    ;       ROUTINE
3500                                    ;       -------
3501  013022  013010                    $TRPAD: .WORD   $TRAP2
3502  013024  011162                            $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
3503  013026  012006                            $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3504  013030  011762                            $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
3505  013032  012022                            $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
```

```
3506  013034  011536                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
3507
3508  013036  012260                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
3509
3510  013040  012210                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
3511  013042  012472                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
3512  013044  012612                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
3513                                  .SBTTL  APT COMMUNICATIONS ROUTINE
3514
3515                                  ;;**************************************************************************
3516  013046  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
3517  013054  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
3518  013062  000403                          BR      $ATYC
3519  013064  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
3520  013072                          $ATYC:
3521  013072  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3522  013074  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3523  013076  105767  000206                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
3524  013102  001450                          BEQ     5$              ;;IF NOT:  BR
3525  013104  122767  000001  165706          CMPB    @APTENV,$ENV    ;;OPERATING UNDER APT?
3526  013112  001031                          BNE     3$              ;;IF NOT:  BR
3527  013114  132767  000100  165677          BITB    @APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
3528  013122  001425                          BEQ     3$              ;;IF NOT:  BR
3529  013124  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
3530  013130  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3531  013136  005767  165636          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
3532  013142  001375                          BNE     1$              ;;IF NOT:  WAIT
3533  013144  010067  165644                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
3534  013150  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
3535  013152  001376                          BNE     2$
3536  013154  166700  165634                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
3537  013160  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
3538  013162  010067  165630                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
3539  013166  012767  000004  165604          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
3540  013174  000413                          BR      5$
3541  013176  017667  000004  000016  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
3542  013204  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
3543  013212  016746  164560                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
3544  013216  004767  175740                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
3545  013222  000000                  4$:     .WORD   0
3546  013224                          5$:
3547  013224  105767  000062          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
3548  013230  001416                          BEQ     12$             ;;IF NOT:  BR
3549  013232  005767  165562                  TST     $ENV            ;;RUNNING UNDER APT?
3550  013236  001413                          BEQ     12$             ;;IF NOT:  BR
3551  013240  005767  165534          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
3552  013244  001375                          BNE     11$             ;;IF NOT:  WAIT
3553  013246  017667  000004  165526          MOV     @4(SP),$FATAL   ;;GET ERROR #
3554  013254  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3555  013262  005267  165512                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
3556  013266  105067  000020          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
3557  013272  105067  000013                  CLRB    $LFLG           ;;CLEAR LOG FLAG
3558  013276  105067  000006                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
3559  013302  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
3560  013304  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
3561  013306  000207                          RTS     PC              ;;RETURN
```

```
3562  013310  000                    $MFLG:  .BYTE   0               ;;MESSG. FLAG
3563  013311  000                    $LFLG:  .BYTE   0               ;;LOG FLAG
3564  013312  000                    $FFLG:  .BYTE   0               ;;FATAL FLAG
3565          013314                          .EVEN
3566          000200                 APTSIZE=200
3567          000001                 APTENV=001
3568          000100                 APTSPOOL=100
3569          000040                 APTCSUP=040
3570                                 ;;**************************************************************************
3571                                 ;THIS ROUTINE WILL INCREMENT THE ERROR COUNT AND THEN PASS THE UNIQUE
3572                                 ;ERROR NUMBER TO THE APT ERROR ROUTINE TO BE REPORTED TO THE APT SYSTEM.
3573
3574  013314  005267  165532         $ERROR: INC     $ERFLG                  ;INCREMENT ERROR FLAG
3575  013320  001775                         BEQ     $ERROR                  ;DON'T LET IT GO TO ZERO
3576  013322  005267  165516                 INC     ERRCNT                  ;INCREMENT THE ERROR COUNT
3577  013326  021627  001002                 CMP     (SP),   #1002           ;IS ERROR FROM VECTOR AREA
3578  013332  101010                         BHI     1$                      ;IF YES THEN
3579  013334  012767  007777  000106         MOV     #7777,  3$              ;REPORT AN UNEXPECTED TRAP
3580  013342  012637  001062                 MOV     (SP)+,@#SAVSP1          ;SAVE UNEXPECTED TRAP DATA
3581  013346  012637  001064                 MOV     (SP)+,@#SAVSP2          ;AND RESTORE SP
3582  013352  000430                         BR      2$                      ;ELSE
3583  013354  017667  000000  000066  1$:    MOV     @(SP),  3$              ;REPORT UNIQUE ERROR NUMBER TO APT
3584  013362  011667  000072                 MOV     (SP),101$               ;SAVE ERROR PC
3585  013366  062716  000002                 ADD     #2,     (SP)            ;GET OVER UNIQUE ERROR NUMBER FOR RETURN
3586  013372  017637  000000  013402  100$:  MOV     @(SP),@#102$
3587  013400  104401                         TYPE                            ;TYPE ERROR MESSAGE
3588  013402  000000                 102$:   .WORD   0
3589  013404  062716  000002                 ADD     #2,(SP)                 ;GET OVER ERROR MESSAGE
3590  013410  104401  001352                 TYPE    ,ERR1                   ;
3591  013414  016746  000030                 MOV     3$,-(SP)                ;PUSH UNIQUE ERROR NUMBER ON THE STACK
3592  013420  104402                         TYPOC                           ;TYPE OCTAL ERROR NUMBER
3593  013422  104401  001366                 TYPE    ,ERR2                   ;
3594  013426  016746  000026                 MOV     101$,-(SP)              ;PUSH ERROR PC ON THE STACK
3595  013432  104402                         TYPOC                           ;TYPE THE ERROR PC
3596  013434  122767  000001  165356  2$:    CMPB    #APTENV,$ENV            ;CHECK TO MAKE SURE WE'RE IN APT MODE
3597  013442  001004                         BNE     5$                      ;IF YES THEN
3598  013444  004767  177414                 JSR     PC,     $ATY4           ;GO REPORT ERROR TO APT
3599  013450  000000                 3$:     .WORD   0                       ;STORAGE FOR ERROR NUMBER
3600  013452  000777                 4$:     BR      4$                      ;LOOP HERE AFTER REPORTING ERROR TO APT
3601  013454  000000                 5$:     HALT                            ;IF NOT APT THEN HALT
3602  013456  000002                         RTI                             ;ALLOW RECOVERY FROM HALT
3603  013460  000000                 101$:   .WORD   0
3604  013462                         $PATCH::
3605  013462  000010                         .BLKW   10
3606          000001                 .END
```

```
ABASE = 000000            625
ACDW1 = 000000            625
ACDW2 = 000000            625
ACPUOP= 000000            625     640
ADDLSB  001070            673#    2537*   2596    2602    2613*   2614
ADDW0 = 000000            625
ADDW1 = 000000            625
ADDW10= 000000            625
ADDW11= 000000            625
ADDW12= 000000            625
ADDW13= 000000            625
ADDW14= 000000            625
ADDW15= 000000            625
ADDW2 = 000000            625
ADDW3 = 000000            625
ADDW4 = 000000            625
ADDW5 = 000000            625
ADDW6 = 000000            625
ADDW7 = 000000            625
ADDW8 = 000000            625
ADDW9 = 000000            625
ADEVCT= 000000            625     631
ADEVM = 000000            625
AENV  = 000000            625     636
AENVM = 000000            625     637
AFATAL= 000000            625     628
ALLCTR  001056            667#    2701*   2705*   2717*   2768*
AMADR1= 000000            625
AMADR2= 000000            625
AMADR3= 000000            625
AMADR4= 000000            625
AMAMS1= 000000            625
AMAMS2= 000000            625
AMAMS3= 000000            625
AMAMS4= 000000            625
AMSGAD= 000000            625     633
AMSGLG= 000000            625     634
AMSGTY= 000000            625     627
AMTYP1= 000000            625
AMTYP2= 000000            625
AMTYP3= 000000            625
AMTYP4= 000000            625
APASS = 000000            625     630
APRIOR= 000000            625
APTCSU= 000040           3084     3569#
APTENV= 000001            896     3077    3525    3567#   3596
APTSFT  001640            896#    1282    1376    1386    1395    1406    1495    1509    1522    1596    1650    1670    1679
                         1699     1791    1816    1894    2175    2194    2217    2234    2252    2261    2360    2395    2405
                         2746     2755
APTSIZ= 000200            945     3566#
APTSPO= 000100           3079     3527    3568#
ASWREG= 002000            544#    625     638
ATESTN= 000000            625     629
AUNIT = 000000            625     632
AUSWR = 000000            625     639
AVECT1= 000000            625
```

```
AVECT2= 000000          625
BACDAT  003052         1269    12940
BEVENT= 177546          5360
BGNTLP  006740         2553#    2607
BIT0   = 000001          357#
BIT00  = 000001          347#     357    1487    1501    1514    1527    1573    1889    1964    2068    2339    2530    2696
BIT01  = 000002          346#     356    2342    2350    2380    2823    2904
BIT02  = 000004          345#     355    1122    1190    1666    1695    2164    2206    2338    2737    2862    2938
BIT03  = 000010          344#     354    1134    1196    1203    1373    1383    1402    1492    1506    1518    1581    1647
                        1663     1676    1692    1788    1813    1891    2258    2357    2392    2740    2854    2925    2956
BIT04  = 000020          343#     353    2191    2231    2249
BIT05  = 000040          342#     352    2172    2214
BIT06  = 000100          341#     351    1786    1809    1811    1966
BIT07  = 000200          340#     350    1368
BIT08  = 000400          339#     349     905    1828    1905    1983    2690    2794
BIT09  = 001000          338#     348    1645    2523    2791
BIT1   = 000002          356#
BIT10  = 002000          337#
BIT11  = 004000          336#
BIT12  = 010000          335#
BIT13  = 020000          334#
BIT14  = 040000          333#
BIT15  = 100000          332#    1486    1490    1500    1503    2803    2818
BIT2   = 000004          355#
BIT3   = 000010          354#
BIT4   = 000020          353#
BIT5   = 000040          352#
BIT6   = 000100          351#
BIT7   = 000200          350#
BIT8   = 000400          349#
BIT9   = 001000          348#
BOARDT= 000001            1#     587     693     730
BPTVEC= 000014          364#
BYPARR  003374         1367    1425#
CACHET= ****** U          1     219     244     587     672     696     709     715     730     751     952     954     969
                        3040    3305
CCR    = 177746         530#     905*    910*    966*   1002*   1052*   1053*   1054    1056    1070   1072*   1122*   1125*
                        1134*   1142*   1190*   1193*   1368*   1418*   1579*   1645*   1704*   1783*   1786*   1809*   1811*
                        1828*   1887*   1889*   1905*   1966*   1968*   1983*   2072*   2074*   2089*   2164*   2167*   2206*
                        2209*   2342*   2350*   2380*   2414*   2531*   2618*   2699*   2823*   2904*   2994*
CKSWR  = 104407         3510#
COUNT   001040          659#
CPEREG= 177766          533#     904*   1007*   1145*   1211*   1289*   1420*   1529*   1603*   1706*   1832*   1908*   1985*
                        2094*   2268*   2418*   2621*   2777*
CPUTST= 000001            1#     244     587     690     730     968    3040
CR     = 000015         272#     709     715     726     730     732     735    3040    3049    3140    3152
CRLF   = 000200         273#    3094    3152
CTST10  003676         1535#
CTST11  004052         1609#
CTST12  004374         1711#
CTST13  004716         1846#
CTST15  005072         1921#
CTST17  005316         2013#
CTST2   002312         1022#
CTST20  005526         2101#
CTST21  006146         2275#
```

```
CTST22  006542          2423#
CTST23  007274          2626#
CTST24  007720          2785#
CTST3   002414          1078#
CTST4   002556          1156#
CTST5   002710          1223#
CTST6   003106          1312#
CTST7   003402          1432#
CURADD  001104           679#  2950*  2951*  2952*  2954
CURDAT  001076           676#
DAPABO  004706          1774   1837#
DATRAM= ****** U         715    953   2423
DCOUNT  001054           666#  2552*  2553   2562   2578   2586   2591*  2718*  2762*
DDISP = 177570           279#   663    926
DISPLA  001050           663#   926*   934*
DISPRE  000174           573#   934
DSWR  = 177570           278#   662    925
EMTVEC= 000030           367#   914*   915*
ENDHRT  003032          1267   1287#
ENDLUP  007222          2614#
ENDTLP  007172          2554   2608#
ERRCNT  001044           661#  3576*
ERRMSG  001325 G         726#   869   1015   1062   1068   1132   1140   1153   1200   1207   1219   1285   1379
                        1389   1398   1409   1416   1428   1498   1512   1525   1599   1653   1673   1682   1702
                        1780   1794   1800   1807   1819   1826   1840   1897   1903   1916   1979   1994   2002
                        2008   2080   2086   2178   2185   2197   2203   2220   2226   2237   2243   2255   2264
                        2365   2371   2400   2408   2560   2572   2751   2760   2849   2858   2873   2886   2921
                        2929   2943   2965
ERRNUM= 000104            543#   866    870#  1013   1016#  1060   1063#  1066   1069#  1130   1133#  1138   1141#
                        1151   1154#  1198   1201#  1205   1208#  1217   1220#  1283   1286#  1377   1380#  1387
                        1390#  1396   1399#  1407   1410#  1414   1417#  1426   1429#  1496   1499#  1510   1513#
                        1523   1526#  1597   1600#  1651   1654#  1671   1674#  1680   1683#  1700   1703#  1778
                        1781#  1792   1795#  1798   1801#  1805   1808#  1817   1820#  1824   1827#  1838   1841#
                        1895   1898#  1901   1904#  1914   1917#  1977   1980#  1992   1995#  2000   2003#  2006
                        2009#  2078   2081#  2084   2087#  2176   2179#  2183   2186#  2195   2198#  2201   2204#
                        2218   2221#  2224   2227#  2235   2238#  2241   2244#  2253   2256#  2262   2265#  2363
                        2366#  2369   2372#  2398   2401#  2406   2409#  2558   2561#  2570   2573#  2749   2752#
                        2758   2761#  2847   2850#  2856   2859#  2871   2874#  2884   2887#  2919   2922#  2927
                        2930#  2941   2944#  2963   2966#
ERROUT  002302          1001   1012#
ERRTN = 000000           541#
ERRVEC= 000004           360#   923    924*   935*
ERR1    001352           730#  3590
ERR2    001366           732#  3593
EXPDAT  001034           657#
EXPTBL  003070          1265   1301#
EXTST   007236          2525   2615   2617#
FLAG    001042           660#  2796*  2798*  2840   2901   2912
FLDO    010134          2838#  2889   2896
FLDOA   010174          2846   2852#
FLD1    010334          2892   2898#
FLD1A   010414          2911#  2979
FLD1B   010454          2918   2923#
FMPARR  002550          1119   1150#
FPPTST= 000001             1#   244    587    688    696    697    730    751    968   3040
FPVEC = 000244           527#
```

L6


```
FSTADD  001100          677#  2546*  2549*  2551  2592
FWDSEQ  001066          672#  2536*  2544  2594  2608  2610*  2612*
GNS   = ******  U       572   3502   3503   3504  3505  3506   3508  3510  3511  3512
GOODAD  001110          681#
GTSWR = 104406          961   3508#
HITMIS= 177752          532#  1004*  1196  1203  1298  1373   1383  1402  1492  1506  1518  1581  1647
                       1656   1676   1685  1788  1813  1891   2172  2191  2214  2231  2249  2258  2357
                       2392   2720   2844  2854  2868  2916   2925  2937
HMPARR  002702         1187   1216#
HT    = 000011         270#   3092   3150
HYBRID= 000001           1#    549    684   691   872   952    1008  1146  1212  1281  1290  1375  1385
                       1394   1405   1421  1494  1508  1521   1530  1595  1604  1649  1669  1678  1698
                       1707   1790   1815  1833  1893  1909   1986  2095  2174  2193  2216  2233  2251
                       2260   2269   2359  2394  2404  2419   2622  2745  2754  2778  3012  3028  3040
                       3054   3305   3513
INITMM  001412          767#  1484   1569  1959  2336  2526   2694  2802
INTERR  005244         1971   1990#
INTRPC  005166         1973#  1998
IOTVEC= 000020          365#   949*   950*
KDPAR0= 172360          508#
KDPAR1= 172362          509#
KDPAR2= 172364          510#
KDPAR3= 172366          511#
KDPAR4= 172370          512#
KDPAR5= 172372          513#
KDPAR6= 172374          514#
KDPAR7= 172376          515#
KDPDR0= 172320          486#
KDPDR1= 172322          487#
KDPDR2= 172324          488#
KDPDR3= 172326          489#
KDPDR4= 172330          490#
KDPDR5= 172332          491#
KDPDR6= 172334          492#
KDPDR7= 172336          493#
KIPAR0= 172340          497#
KIPAR1= 172342          498#
KIPAR2= 172344          499#
KIPAR3= 172346          500#
KIPAR4= 172350          501#  2805*
KIPAR5= 172352          502#  2838*  2842*  2911*  2914*
KIPAR6= 172354          503#  1570*  1961*  2066*  2337*  2529*  2698*  2704*  2713*  2767*  2817*  2899*  2903*
KIPAR7= 172356          504#
KIPDR0= 172300          475#  1486*  1490*  1500*  1503*  2818*
KIPDR1= 172302          476#
KIPDR2= 172304          477#
KIPDR3= 172306          478#
KIPDR4= 172310          479#
KIPDR5= 172312          480#  2803*
KIPDR6= 172314          481#
KIPDR7= 172316          482#
LDPARS  001554          768    770    772   774   776   778    8120#
LDPDRS  001604          780    782    784   786   788   790    8410#
LF    = 000012          271#   709    715   726   730   732    735   3040  3049  3144  3152
LOOPIN  001060          668#  1263*  1266  1278*  2697*  2771*
LOWADD  001106          680#  2693*  2700  2711  2770*
```

```
LSTADD  001102          678#  2547*  2550*  2574
MMUTST= 000001            1#   244    587    687    709    730    751    952    954    968   3040   3305
MMVEC  = 000250          376#
MSER   = 177744          531#   902*  1003*  1006*  1144*  1210*  1288*  1419*  1528*  1602*  1705*  1775*  1776
                        1796   1802*  1803   1822   1831*  1886*  1907*  1969*  1984*  2004   2093*  2267*  2417*
                        2620*  2776*
MSG1    011054          3014   3040#
MSG2    011135          3024   3049#
MXVDEL= 000001            1#   1921   2013
NEWADD  001114          683#
NEWDAT  001074          675#
OPMSG2  001162          709#   952
OPMSG3  001224          715#   953
PARRAM= 000001            1#   2013
PIRQ   = 177772          277#
PIRQVE= 000240          371#
PR0    = 000000          294#
PR1    = 000040          295#
PR2    = 000100          296#
PR3    = 000140          297#
PR4    = 000200          298#
PR5    = 000240          299#
PR6    = 000300          300#
PR7    = 000340          301#
PS     = 177776          274#   275
PSW    = 177776          275#
PWRVEC= 000024          366#
QBUSEX= 000001            1#   689    3001
RAMPAR  001630          866#   999    1262   1483   2349   2539   2715
RBUF   = 177562          538#
RCSR   = 177560          537#
RDCHR  = 104410         3418    3511#
RDLIN  = 104411         3512#
RECDAT  001036          658#   2366*  2367   2401*  2402
RESTAR  002176          965#   3037
RESVEC= 000010          361#
RITEDA  001072          674#
SAVSP1  001062          669#   3580*
SAVSP2  001064          670#   3581*
SDPAR0= 172260          464#
SDPAR1= 172262          465#
SDPAR2= 172264          466#
SDPAR3= 172266          467#
SDPAR4= 172270          468#
SDPAR5= 172272          469#
SDPAR6= 172274          470#
SDPAR7= 172276          471#
SDPDR0= 172220          442#
SDPDR1= 172222          443#
SDPDR2= 172224          444#
SDPDR3= 172226          445#
SDPDR4= 172230          446#
SDPDR5= 172232          447#
SDPDR6= 172234          448#
SDPDR7= 172236          449#
SHMOD  = 000001           1#   614    3040
```

```
SIPAR0= 172240          453#
SIPAR1= 172242          454#
SIPAR2= 172244          455#
SIPAR3= 172246          456#
SIPAR4= 172250          457#
SIPAR5= 172252          458#
SIPAR6= 172254          459#
SIPAR7= 172256          460#
SIPDR0= 172200          431#
SIPDR1= 172202          432#
SIPDR2= 172204          433#
SIPDR3= 172206          434#
SIPDR4= 172210          435#
SIPDR5= 172212          436#
SIPDR6= 172214          437#
SIPDR7= 172216          438#
SKIPID  010760         3013    3015#
SLOC00  001030          653#    998#  1773#  1829   1884#  1906   1970#  1981   2064#  2091   2348#  2416   2538#
                       2617   2714#  2775   2799#  2997
SLOC01  001032          654#   2998
SOFTER  001116          684#    898    901#  1008#  1146#  1212#  1290#  1421#  1530#  1604#  1707#  1833#  1909#
                       1986#  2095#  2269#  2419#  2622#  2778#
SOFTRE  001120          685#    906   1281#  1375#  1385#  1394#  1405#  1494#  1508#  1521#  1595#  1649#  1669#
                       1678#  1698#  1790#  1815#  1893#  2174#  2193#  2216#  2233#  2251#  2260#  2359#  2394#
                       2404#  2745#  2754#
SR0     = 177572        380#    903#  1487#  1489#  1501#  1504#  1514#  1520#  1527#  1573#  1590#  1601#  1964#
                       1982#  2068#  2083#  2092#  2339#  2415#  2530#  2619#  2696#  2774#  2809#  2995#
SR1     = 177574        381#
SR2     = 177576        382#
SR3     = 172516        383#  2067#  2338#  2695#  2804#
STACK   = 001000        265#   912
START   001716          570    576    909#
STBOT   = 001000        548#  2996
STKLMT= 177774          276#
SWR     001046          662#   925#   927    933#   947#   959   2523   2690   2791   2794   3315   3352#
SWREG   000176          574#   933    959   3315   3328
SW0     = 000001        329#
SW00    = 000001        319#   329
SW01    = 000002        318#   328
SW02    = 000004        317#   327
SW03    = 000010        316#   326
SW04    = 000020        315#   325
SW05    = 000040        314#   324
SW06    = 000100        313#   323
SW07    = 000200        312#   322
SW08    = 000400        311#   321
SW09    = 001000        310#   320
SW1     = 000002        328#
SW10    = 002000        309#
SW11    = 004000        308#
SW12    = 010000        307#
SW13    = 020000        306#
SW14    = 040000        305#
SW15    = 100000        304#
SW2     = 000004        327#
SW3     = 000010        326#
```

```
SW4     = 000020            3250
SW5     = 000040            3240
SW6     = 000100            3230
SW7     = 000200            3220
SW8     = 000400            3210
SW9     = 001000            3200
TAGAD1    010672            2824    2895    2983#
TAGAD2    010704            2908    2978    2988#
TAGRAM=   ******  U          715     953    2776    2781    2783
TAPAB0    005062            1885   1913#
TBITVE=   000014             362#
TKVEC   = 000060             369#
TPVEC   = 000064             370#
TRAPVE=   000034             368#    916*    917*
TRTVEC=   000014             363#
TSEND     010716            2980   2994#
TSTADD    001112             682#
TSTLOC    001122             695#   1120*   1123*   1127    1135    1143*   1188*   1191*   1195    1202    1270    1273    1275
                            1369    1371*   1372    1381*   1382*   1392    1400*   1401*   1412    1485*   1488*   1491    1502
                            1505    1515    1643    1644    1646    1655*   1658    1675    1684*   1687    1782*   1785*   1787
                            1810*   1812    1888*   1890    1962*   1967*   1973    2163*   2165*   2170    2188    2207*   2212
                            2229    2246    2248*   2257
TSTLUP    006656            2541#   2611    2616
TST1      002210             996#
TST10     003676            1531   1567#   1595
TST11     004052            1605   1641#   1649    1669    1678    1698
TST12     004374            1708   1771#   1790    1815
TST13     004716            1834   1878#   1893
TST14     005072            1910   1957#
TST15     005316            1987   2061#
TST16     005526            2096   2161#   2174    2193    2216    2233    2251    2260
TST17     006146            2270   2334#   2359    2394    2404
TST2      002312            1009   1049#
TST20     006542            2420   2521#
TST21     007274            2623   2688#   2745    2754
TST22     007720            2779   2789#
TST3      002414           1117#
TST4      002556            1147   1185#
TST5      002710            1213   1260#   1281
TST6      003106            1291   1365#   1375    1385    1394    1405
TST7      003402            1422   1481#   1494    1508    1521
TYPDS   = 104405            3026   3506#
TYPE    = 104401             952    953    3014    3024    3027    3097    3219    3288    3326    3327    3330    3343    3354
                            3373   3422    3425    3429   3502#    3587    3590    3593
TYPOC   = 104402            3329   3503#   3592    3595
TYPON   = 104404           3505#
TYPOS   = 104403           3504#
UDPAR0= 177660              420#
UDPAR1= 177662              421#
UDPAR2= 177664              422#
UDPAR3= 177666              423#
UDPAR4= 177670              424#
UDPAR5= 177672              425#
UDPAR6= 177674              426#
UDPAR7= 177676              427#
UDPDR0= 177620              398#
```

```
UDPDR1= 177622            3990
UDPDR2= 177624            4000
UDPDR3= 177626            4010
UDPDR4= 177630            4020
UDPDR5= 177632            4030
UDPDR6= 177634            4040
UDPDR7= 177636            4050
UIPAR0= 177640            4090
UIPAR1= 177642            4100
UIPAR2= 177644            4110
UIPAR3= 177646            4120
UIPAR4= 177650            4130
UIPAR5= 177652            4140
UIPAR6= 177654            4150
UIPAR7= 177656            4160
UIPDR0= 177600            3870
UIPDR1= 177602            3880
UIPDR2= 177604            3890
UIPDR3= 177606            3900
UIPDR4= 177610            3910
UIPDR5= 177612            3920
UIPDR6= 177614            3930
UIPDR7= 177616            3940
XBUF  = 177566            5400
XCSR  = 177564            5390
XXX     010732           2793    29970
$APTHD  000204            596      6020
$ASTAT= ****** U          3547    3562
$ATYC   013072            3518    35200
$ATY1   013046            35160
$ATY3   013054            3082    35170
$ATY4   013064            35190   3598
$AUTOB  012764            9630    3323    34440
$CHARC  011512            3099*   3109*   3116    3142*   31470
$CKSWR  012210            33150   3510
$CMTAG= ****** U           912     918     920
$CNTLG  012735            3326    34390
$CNTLU  012730            3343    34380
$CPUOP  001026             6400
$CRLF   001403             7350    3098    3159    3354    3438
$DBLK   011752            3185    3219    32270
$DEVCT  001010             6310
$DOAGN  011044            3020    3029    30350
$DTBL   011742            3188    32230
$ENDAD  011034             583    30310
$ENDCT  011002             919    30220
$ENULL  011050            3027    30380
$ENV    001020             6360    896     957     3077    3525    3549    3596
$ENVM   001021             6370    945    3079    3084    3527
$EOP    010746            30110
$EOPCT  010774             919*   30190   3023
$ERFLG  001052             6640    920*   3574*
$ERROR  013314             914     949    35740   3575
$ETABL  001020             6350
$ETEND  001030             608     6470
$FATAL  001002             6280   3553*
```

```
$FFLG   013312          3516* 3519* 3547  3556* 3564#
$FILLC  011530          3102  3156#
$FILLS  011527          3155#
$GET42  011024          3028#
$GTSWR  012260          3327# 3508
$HD   = 000003           253   254
$HIBTS  000204           603#
$INTAG  012765          3355  3445#
$LF     011533          3159# 3429  3438
$LFLG   013311          3557* 3563#
$MAIL   001000           604   608   626#  937   957   3077
$MBADR  000206           604#
$MFLG   013310          3517* 3523  3558* 3562#
$MNEW   012753          3330  3442#
$MSGAD  001014           633#  3533* 3536
$MSGLG  001016           634#  3538*
$MSGTY  001000           627#  3531  3539* 3551  3555*
$MSWR   012742          3327  3440#
$NULL   011526          3104  3154#
$NWTST= 000001           969#  971   1023# 1025  1079# 1081  1157# 1159  1224# 1226  1313# 1315  1433#
                        1435  1536# 1538  1610# 1612  1712# 1714  1847# 1849  1922# 1924  2014# 2016
                        2102# 2104  2276# 2278  2424# 2426  2627# 2629  2786#
$OCNT   012204          3260* 3289* 3302#
$OMODE  012206          3255* 3259* 3264  3267* 3278* 3304#
$PASS   001006           630#  918*  944* 3012  3016* 3017* 3025  3038
$PASTM  000212           606#
$PATCH  013462 G        3604#
$QUES   011532          3158# 3373  3422  3438
$RDCHR  012472          3386# 3511
$RDDEC= ****** U        3513
$RDLIN  012612          3414# 3512
$RDOCT= ****** U        3513
$RDSZ = 000010          3407#
$RTNAD  011046          3037#
$R2A  = ****** U        3513
$SAVRE= ****** U        3513
$SETUP= 000126           549#  913   914   916   918   920   921   954   3016  3310  3444
$STUP = 177777           549#
$SVPC = 000204           581#  586
$SWR  = 160000           253   254#  921   998   1051  1119  1187  1262  1367  1483  1569  1643  1773
                        1880  1959  2063  2163  2336  2523  2690  2791  3008  3016  3030  3036  3038
$SWREG  001022           638#  947
$TESTN  001004           629#  965*  997*  1050* 1118* 1186* 1261* 1366* 1482* 1568* 164?* 1772* 1879*
                        1958* 2062* 2162* 2335* 2522* 2689* 2790*
$TKB    011520          3123  3130  3151# 3308  3319  3336  3390  3396
$TKS    011516          3121  3128  3150# 3308  3317  3333  3357* 3388  3394
$TN   = 000023           253#  969   998#  1006  1023  1051# 1079  1119# 1144  1157  1187# 1209  1224
                        1262# 1281  1287  1313  1367# 1375  1385  1394  1405  1419  1433  1483# 1494
                        1508  1521  1528  1536  1569# 1595  1602  1610  1643# 1649  1669  1678  1698
                        1705  1712  1773# 1790  1815  1830  1847  1880# 1893  1907  1922  1959# 1984
                        2014  2063# 2093  2102  2163# 2174  2193  2216  2233  2251  2260  2266  2276
                        2336# 2359  2394  2404  2417  2424  2523# 2620  2627  2690# 2745  2754  2776
                        2786  2791#
$TPB    011524          3139* 3153#
$TPFLG  011531          3071  3157#
$TPS    011522          3137  3152#
```

```
$TRAP   012766          916    3454#
$TRAP2  013010         3465#   3501
$TRP  = 000012         3494#   3503#  3504#  3505#  3506#  3507#  3508   3509#  3510   3511#  3512#  3513#
$TRPAD  013022         3459    3501#
$TSTM   000210          605#
$TSTNU= 000001          542#
$TTYIN  012720         3415    3416   3433   3437#
$TYPBN= ****** U       3507
$TYPDS  011536         3173#   3506
$TYPE   011162         3071#   3494   3502   3544
$TYPEC  011374         3101    3108   3115   3120#  3359
$TYPEX  011514         3143    3145   3148#
$TYPOC  012006         3258#   3503
$TYPON  012022         3257    3260#  3505
$TYPOS  011762         3253#   3504
$UNIT   001012          632#
$UNITM  000214          607#
$USWR   001024          639#
$XOFF = 000023         3125    3152
$XON  = 000021         3132    3152
$$GET4= 000000         3030#
$OFILL  012205         3254*   3258*  3268   3303#
$40CAT  000000          567#
.     = 013502          552#    568    572#   581    582#   584#   586#   592    593#   595#   597#   609#   610#
                        614#    615#   696#   737#   3038   3039#  3053#  3150   3151   3152   3153   3154   3155
                       3156    3157   3158   3159   3160#  3227#  3308   3437#  3438   3444   3445   3446   3565#
                       3605#
.$ASTA= ****** U       3517    3520
.$X   = 000204          592#    597    609    614
```

```
A         1281#  1375#  1385#  1394#  1405#  1494#  1508#  1521#  1595#  1649#  1669#  1678#  1698#  1790#  1815#
          1893#  2174#  2193#  2216#  2233#  2251#  2260#  2359#  2394#  2404#  2745#  2754#
BGNMOD     238#   967
BGNSUB     239#
BGNTST     238#   996   1049   1117   1185   1260   1365   1481   1567   1641   1771   1878   1957   2061   2161
          2334   2521   2688   2789
CCHMS2    1021#  1025
CCHMS3    1078#  1081
CCHMS4    1156#  1159
CCHMS5    1223#  1226
CCHMS6    1311#  1315
CKLOOP     238#
CMSG1      969#   971
CMSG10    1535#  1538
CMSG11    1609#  1612
CMSG12    1711#  1714
CMSG13    1846#  1849
CMSG15    1921#  1924
CMSG17    2013#  2016
CMSG20    2101#  2104
CMSG21    2275#  2278
CMSG22    2423#  2426
CMSG23    2626#  2629
CMSG7     1432#  1435
COMMEN     372#
DEFPRG      1#
ENDCOM     372#
ENDMOD     238#  3000
ENDPAS    3002#  3024
ENDSUB     239#
ENDTST     238#  1019   1076   1155   1221   1309   1430   1533   1607   1710   1844   1920   2012   2098   2272
          2422   2625   2781   2999
ERRDEF     237#
ERRDF      238#   866   1013   1060   1066   1130   1138   1151   1198   1205   1217   1283   1377   1387   1396
          1407   1414   1426   1496   1510   1523   1597   1651   1671   1680   1700   1778   1792   1798   1805
          1817   1824   1838   1895   1901   1914   1977   1992   2000   2006   2078   2084   2176   2183   2195
          2201   2218   2224   2235   2241   2253   2262   2363   2369   2398   2406   2558   2570   2749   2758
          2847   2856   2871   2884   2919   2927   2941   2963
ERRF01     235#
ERRF02     235#
ERRF03     235#  2174
ERRF04     235#  2193
ERRF05     236#  2216
ERRF06     236#  2233
ERRF07     236#
ERRF08     236#  2359
ERRF09     236#  2394
ERRF1      227#  1012
ERRF10     228#  1385
ERRF11     228#  1394
ERRF12     228#  1405
ERRF13     228#  1414
ERRF14     228#  1494
ERRF15     228#  1508
ERRF16     229#  1521
ERRF17     229#  1595
```

ERRF18 2290 1649
ERRF19 2290 1668
ERRF2 2270 1060
ERRF20 2290 1678
ERRF21 2290 1697
ERRF22 2290 1778
ERRF23 2290 1790
ERRF24 2300 1798
ERRF25 2300 1805
ERRF26 2300 1815
ERRF27 2300 1824
ERRF28 2300
ERRF29 2300 1837
ERRF3 2270 1065
ERRF30 2300 1893
ERRF31 2300 1900
ERRF32 2310 1913
ERRF33 2310
ERRF34 2310
ERRF35 2310
ERRF36 2310
ERRF37 2310 1977
ERRF38 2310 1992
ERRF39 2310 2000
ERRF4 2270 1130
ERRF40 2320 2006
ERRF41 2320
ERRF42 2320
ERRF43 2320
ERRF44 2320
ERRF45 2320
ERRF46 2320 2078
ERRF47 2320 2084
ERRF48 2330 2182
ERRF49 2330 2200
ERRF5 2270 1138
ERRF50 2330 2223
ERRF51 2330 2240
ERRF52 2330 2251
ERRF53 2330 2260
ERRF54 2330 2369
ERRF55 2330 2404
ERRF56 2340 2558
ERRF57 2340 2570
ERRF58 2340 2745
ERRF59 2340 2754
ERRF6 2270 1198
ERRF60 2340 2847
ERRF61 2340 2856
ERRF62 2340 2871
ERRF63 2340 2884
ERRF64 2350 2919
ERRF65 2350 2927
ERRF66 2350 2941
ERRF67 2350 2963
ERRF7 2270 1205

```
ERRF8     2280#  1280
ERRF9     2280#  1375
ERROR     2660#   867  1013  1060  1066  1130  1138  1151  1198  1205  1217  1283  1377  1387  1396
          1407   1414  1426  1496  1510  1523  1597  1651  1671  1680  1700  1778  1792  1798  1805
          1817   1824  1838  1895  1901  1914  1977  1992  2000  2006  2078  2084  2176  2183  2195
          2201   2218  2224  2235  2241  2253  2262  2363  2369  2398  2406  2558  2570  2749  2758
          2847   2856  2871  2884  2919  2927  2941  2963
ESCAPE    3720#
ETXT1     2190#   992
ETXT10    2200#  1342
ETXT11    2200#  1346
ETXT12    2200#  1351
ETXT13    2200#  1355
ETXT14    2200#  1461
ETXT15    2200#  1469
ETXT16    2200#  1475
ETXT17    2200#  1562
ETXT18    2210#  1623
ETXT19    2210#  1627
ETXT2     2190#  1039
ETXT20    2210#  1632
ETXT21    2210#  1636
ETXT22    2210#  1729
ETXT23    2210#  1738
ETXT24    2210#  1741
ETXT25    2210#  1746
ETXT26    2220#  1753
ETXT27    2220#  1756
ETXT28    2220#  1761
ETXT29    2220#  1766
ETXT3     2190#  1042
ETXT30    2220#  1866
ETXT31    2220#  1868
ETXT32    2220#  1874
ETXT33    2220#
ETXT34    2230#
ETXT35    2230#
ETXT36    2230#
ETXT37    2230#  1937
ETXT38    2230#  1943
ETXT39    2230#  1948
ETXT4     2190#  1103
ETXT40    2230#  1951
ETXT41    2230#
ETXT42    2240#
ETXT43    2240#
ETXT44    2240#
ETXT45    2240#
ETXT46    2240#  2047
ETXT47    2240#  2057
ETXT48    2240#  2135
ETXT49    2240#  2139
ETXT5     2190#  1110
ETXT50    2250#  2144
ETXT51    2250#  2148
ETXT52    2250#  2153
```

```
ETXT53    2250#  2157
ETXT54    2250#  2308
ETXT55    2250#  2325
ETXT56    2250#  2461
ETXT57    2250#  2470
ETXT58    2260#  2668
ETXT59    2260#  2672
ETXT6     2190#  1176
ETXT60    2260#
ETXT61    2260#
ETXT62    2260#
ETXT7     2190#  1180
ETXT8     2190#  1244
ETXT9     2190#  1337
EXITST    2360#  1006  1144  1209  1287  1419  1528  1602  1705  1830  1907  1984  2093  2266  2417
          2620   2776
FRONT       1#
FRONT1      1#   2160#
GETPRI    3720#
GETSWR    3720#   954
IDMSG    30020#  3012
MULT      3720#
NEWTST    2370#  3720#   969  1022  1078  1156  1223  1312  1432  1535  1609  1711  1846  1921  2013
          2101   2275  2423  2626  2785
POP       3720#  3214  3559  3560
PUSH      3720#  3173  3520  3522  3543
REPORT    3720#
SCOPE     2670#
SETPRI    3720#
SETTRA    3469#  3494  3503  3504  3505  3506  3508  3510  3511  3512
SETUP     2370#   3720#  911
SFTERR    2360#  1281  1375  1385  1394  1405  1494  1508  1521  1595  1649  1669  1678  1698  1790
          1815   1893  2174  2193  2216  2233  2251  2260  2359  2394  2404  2745  2754
SKIP      3720#
SLASH     3720#
SPACE     3720#
STARS     3720#   579   589   591   598   624   969   971   995  1023  1025  1048  1079  1081  1116
          1157   1159  1184  1224  1226  1259  1313  1315  1364  1433  1435  1480  1536  1538  1566
          1610   1612  1640  1712  1714  1770  1847  1849  1877  1922  1924  1956  2014  2016  2060
          2102   2104  2160  2276  2278  2333  2424  2426  2520  2627  2629  2687  2786  2788  3005
          3056   3163  3230  3307  3310  3378  3407  3448  3515  3570
SWRSU     3720#  9210#
TAIL        1#   3002
TRMTRP    3491#
TYPBIN    3720#
TYPDEC    3720#  3025
TYPNAM    3720#
TYPNUM    3720#
TYPOCS    3720#
TYPOCT    3720#  3328
TYPTXT    3720#
$$ESCA    3720#
$$FTER   1281#  1375#  1385#  1394#  1405#  1494#  1508#  1521#  1595#  1649#  1669#  1678#  1698#  1790#  1815#
         1893#  2174#  2193#  2216#  2233#  2251#  2260#  2359#  2394#  2404#  2745#  2754#
$$NEWT    3720#   969  1023  1079  1157  1224  1313  1433  1536  1610  1712  1847  1922  2014  2102
          2276   2424  2627  2786
```

```
$$SET    3475#   3494    3503    3504    3505    3506    3508    3510    3511    3512
$$SETM    937#    944
$$SKIP    372#
.EQUAT    237#    262
.HEADE    239#    244
.KT11     237#    372
.SETUP    239#    549
.$ACT1    239#    577
.$APTB    237#    622
.$APTH    239#    587
.$APTY    240#   3513
.$EOP     237#   3003
.$ERRO    240#
.$READ    240#   3305
.$TRAP    239#   3446
.$TYPD    238#   3161
.$TYPE    238#   3054
.$TYPO    240#   3228
.$40CA    237#    550


. ABS.  013502     000

 ERRORS DETECTED:  0

 CZKDMB/EN:ABS,CZKDMB.SEQ/DOC/SOL/CRF/NL:TOC=SYSMAC.SML/ML,CZKDMB.MAC/ML,KDJ11A.MAC
 RUN-TIME:  748 109 5 SECONDS
 RUN-TIME RATIO: 1286/864=1.4
 CORE USED:  58K  (115 PAGES)
 DOCUMENT PAGES:  87
```